

RECOMENDACIÓN DINÁMICA DE PRODUCTOS A GRUPOS CON REALIDAD AUMENTADA

Cristina Delgado Rodríguez

Ignacio Rocillo Landa

Jorge Muñoz Rodríguez

Jorge Rueda Garzón

Sergio Fuentes Urabayen



TRABAJO FIN DE GRADO

FACULTAD DE INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

2015-2016

*A todos nuestros familiares,
que siempre nos han estado dando apoyo
durante la realización de este proyecto.*

AUTORIZACIÓN

Nosotros, Cristina Delgado Rodríguez, Ignacio Rocillo Landa, Jorge Muñoz Rodríguez, Jorge Rueda Garzón y Sergio Fuentes Urabayen, alumnos matriculados en la asignatura Trabajo de Fin de Grado (TFG) en la Facultad de Informática de la Universidad Complutense de Madrid durante el curso 2015/2016, dirigidos por Juan Antonio Recio García y Guillermo Jiménez Díaz, autorizamos la difusión y utilización con fines académicos, no comerciales, y mencionando expresamente a sus autores del contenido de esta memoria, el código, la documentación adicional y la aplicación desarrollada.

Cristina Delgado Rodríguez, Ignacio Rocillo Landa, Jorge Muñoz Rodríguez, Jorge Rueda Garzón y Sergio Fuentes Urabayen

AGRADECIMIENTOS

En primer lugar, quisiéramos agradecer a todos nuestros familiares que durante la realización de este proyecto siempre nos han apoyado y que han puesto toda su ilusión en nuestro viaje universitario.

Mencionar también a todas las personas que han evaluado MUVI y han dedicado una parte de su tiempo a valorarla y probarla, especialmente a nuestras familias y amigos.

Por último, quisiéramos dar las gracias a nuestros tutores de Trabajo de Fin de Grado, Juan Antonio Recio García y Guillermo Jiménez Díaz, por permitirnos trabajar en este proyecto y ayudarnos y guiarnos en la realización de este TFG.

RESUMEN

Resumen

La Realidad Aumentada es una tecnología que permite aumentar el mundo real que percibimos con elementos virtuales interactivos. En esta memoria describimos el uso de esta tecnología, entre otras, para obtener información a tiempo real sobre películas.

La aplicación que describimos es capaz de recoger toda la información de una película con solo enfocar su foto de portada con la cámara, pudiendo guardar y/o compartir esta información. Además, explicaremos el sistema de recomendación para grupos de personas, que también es una funcionalidad de nuestra aplicación. Este sistema recoge las valoraciones de todos los usuarios para luego hacer una recomendación grupal.

Veremos de una manera detallada cómo ha sido el proceso evolutivo desde la idea inicial hasta llegar a una aplicación real.

Abstract

Augmented Reality is a technology which allows to increase the real world that we sense with interactive virtual elements. In this report, we describe the use of this technology, among others, for obtaining information about movies.

The application described is able to collect all information of a film with only focus your cover photo with the camera and can store and / or share this information. In addition, we will explain the recommendation system for groups of people, which is also a feature of our application. This system recollects the evaluations of all users and then makes a recommendation group.

We will see in a detailed way how it has been the evolutionary process from the initial idea to reach a real application.

INDICE

AUTORIZACIÓN.....	III
AGRADECIMIENTOS	IV
RESUMEN	V
RESUMEN	V
ABSTRACT	V
INDICE	I
INDICE DE FIGURAS.....	IV
CAPÍTULO 1: INTRODUCCIÓN A MUVI.....	1
1.1 INTRODUCCIÓN	1
1.2 OBJETIVOS	2
1.3 ORGANIZACIÓN DE LA MEMORIA.....	2
CAPÍTULO 2: ESTADO DEL ARTE	4
2.1 REALIDAD AUMENTADA	4
2.1.1 Usos.....	5
2.1.2 Tecnologías	8
2.2 FUENTES DE INFORMACIÓN DE PELÍCULAS	10
2.2.1 Metacritic.....	10
2.2.2 IMDb	11
2.2.3 Rotten Tomatoes.....	12
2.2.4 TheMovieDb.Org	13
2.3 SISTEMAS DE RECOMENDACIÓN	14
2.3.1 Tipos.....	14
2.3.2 Tecnologías	17
2.4 SISTEMAS DE LOCALIZACIÓN CON DISPOSITIVOS BEACON	18
2.4.1 Usos.....	18
2.4.2 Tecnologías	19
CAPÍTULO 3: FUNCIONALIDAD	22
3.1 FUNCIONES DE LA APLICACIÓN	22
3.2 FLUJO DE LA APLICACIÓN	22
3.3 PRIMEROS PROTOTIPOS	24
3.4 INTERFAZ DE USUARIO	26
3.4.1 Interfaz de Registro	26
3.4.2 Interfaz de Iniciar Sesión	27
3.4.3 Interfaz de Inicio.....	28
3.4.4 Interfaz de Realidad Aumentada	30
3.4.5 Interfaz de Me Gusta	35
3.4.6 Interfaz de Ficha de Película.....	36
3.4.7 Interfaz de Gente Cercana	38
CAPÍTULO 4: IMPLEMENTACIÓN	40
4.1 ARQUITECTURA.....	40
4.1.1 Relación entre front-end y back-end.....	41
4.2 IMPLEMENTACIÓN DEL BACK-END	42
4.2.1 Arquitectura de servicios.....	43
4.2.2 Base de datos	46
4.2.3 Sistema de recomendación	48

4.2.4 Servidor	50
4.2.5 Fuente de información sobre películas.....	50
4.3 IMPLEMENTACIÓN DEL FRONT-END	51
4.3.1 Entorno de desarrollo.....	52
4.3.2 Realidad Aumentada.....	53
4.3.3 Tecnologías	54
4.3.4 Herramientas	57
4.4 PRUEBAS DE ARQUITECTURA	58
4.4.1 Pruebas de realidad aumentada.....	58
4.4.2 Pruebas de servicios web REST con Jersey	63
4.4.3 Pruebas con Hibernate/JPA.....	64
4.4.4 Pruebas de los servicios de redes sociales.....	65
4.4.5 Pruebas de geolocalización	68
CAPÍTULO 5: EVALUACIÓN DE USUARIOS	70
5.1 PLAN DE EVALUACIÓN.....	70
5.1.1 ¿Qué estamos evaluando?.....	70
5.1.2 Propósito de la evaluación	70
5.1.3 Objetivos generales.....	71
5.1.4 Preguntas de investigación	71
5.1.5 Requisitos de los participantes.....	71
5.1.6 Diseño experimental	71
5.1.7 Tareas a realizar.....	72
5.1.8 Entorno y herramientas empleadas.....	72
5.1.9 Obtención de feedback de los participantes	73
5.1.10 Tareas del moderador.....	73
5.1.11 Descripción de la metodología de análisis de datos	73
5.2 EVALUACIÓN	74
5.2.1 Observaciones y opiniones de los usuarios	74
5.2.2 Resultado cuestionario SUS.....	76
CAPÍTULO 6: CONCLUSIONES	78
6.1 CONCLUSIONES	78
6.2 CONCLUSIONS.....	80
CAPÍTULO 7: TRABAJO FUTURO	83
CAPÍTULO 8: ORGANIZACIÓN DEL TRABAJO.....	85
8.1 MODELO DE DESARROLLO	88
8.2 HERRAMIENTAS DE COMUNICACIÓN	88
8.3 HERRAMIENTAS DE CONTROL DE VERSIONES.....	88
8.4 HERRAMIENTAS DE GESTIÓN DE TAREAS	89
CAPÍTULO 9: APORTACIONES AL PROYECTO	90
9.1 CRISTINA DELGADO RODRÍGUEZ	90
9.2 IGNACIO ROCILLO LANDA	93
9.3 JORGE MUÑOZ RODRÍGUEZ.....	94
9.4 JORGE RUEDA GARZÓN.....	96
9.5 SERGIO FUENTES URABAYEN.....	98
REFERENCIAS	100
ANEXOS	103
ANEXO I: API REST.....	103
1. Usuarios	103
2. Películas.....	105

3.	<i>Valoraciones</i>	108
4.	<i>Deseos</i>	109
5.	<i>Recomendaciones</i>	111
6.	<i>Geolocalización (GPS)</i>	111
7.	<i>Beacons</i>	113

INDICE DE FIGURAS

Figura 1: Ejemplo de uso en la arqueología	5
Figura 2: Ejemplo de uso en la arquitectura	5
Figura 3: Ejemplo de uso en la medicina	6
Figura 4: Ejemplo de uso militar	6
Figura 5: Ejemplo de uso en museo	7
Figura 6: Ejemplo de uso en los videojuegos	7
Figura 7: Ejemplo de Metacritic	10
Figura 8: Ejemplo de IMDb	11
Figura 9: Ejemplo de Rotten Tomatoes	12
Figura 10: Ejemplo de Rotten Tomatoes	13
Figura 11: Filtrado colaborativo basado en el usuario	14
Figura 12: Filtrado colaborativo basado en el contenido	15
Figura 13: Funcionalidad de los dispositivos Beacon dentro de un centro comercial	18
Figura 14: Ilustración de Seguridad	19
Figura 15: Ilustración y aplicación de los Estimote Beacons	20
Figura 16: Ilustración de los diferentes dispositivos de RadBeacon	20
Figura 17: Partes de que se compone Bluecat Beacon	21
Figura 18: Ilustración del modelo Kontakt.io	21
Figura 19: Flujo de las interfaces de la aplicación	23
Figura 20: Prototipo de interfaz de búsqueda Figura 21: Ilustración del modelo Kontakt.io	24
Figura 22: Primer prototipo de la interfaz de inicio	25
Figura 23: Segundo prototipo de la interfaz de inicio	25
Figura 24: Interfaz de registro	26
Figura 25: Diagrama de actividad de registro	27
Figura 26: Interfaz de inicio de sesión	27
Figura 27: Diagrama de actividad de la interfaz de inicio de sesión	28
Figura 28: Interfaz de inicio y menú lateral	28
Figura 29: Diagrama de actividad de inicio	29
Figura 30: Interfaz de RA	30
Figura 31: Trailer en la interfaz de RA	31
Figura 32: Valoración en RA	31
Figura 33: Información de la película en RA	32
Figura 34: Información de la película en RA	32
Figura 35: Información de la película en RA	33
Figura 36: Amigos cercanos en RA	34

Figura 37: Diagrama de actividad de la RA	34
Figura 38: Interfaz de Me Gusta	35
Figura 39: Diagrama de actividad de Me Gusta	36
Figura 40: Interfaz de ficha de película	36
Figura 41: Diagrama de actividad de Ficha de Película	37
Figura 42: Interfaz de Gente Cercana	38
Figura 43: Diagrama de actividad de Gente Cercana	39
Figura 44: Arquitectura de la aplicación.....	40
Figura 45: Módulos del back-end.....	42
Figura 46: Diseño E/R de la base de datos MySQL	47
Figura 47: Módulos del front-end	52
Figura 48: Ejemplo de uso de los fragmentos de Android	55
Figura 49: Prueba de reconocimiento local con Vuforia	59
Figura 50: Prueba de reconocimiento en la nube con Vuforia.....	60
Figura 51: Prueba de reconocimiento en local con Wikitude.....	61
Figura 52: Prueba de reconocimiento en la nube con Wikitude	63
Figura 53: Prueba de saludo.....	63
Figura 54: Prueba de suma.....	64
Figura 55: Ejemplo de autorización de Twitter.....	65
Figura 56: Ejemplo de tweet hecho desde la app.....	65
Figura 57: Ejemplo de publicación en Facebook.....	67
Figura 58: Ejemplo de botón de Facebook	67
Figura 59: Segundo ejemplo de publicación en Facebook.....	67
Figura 60: Prueba de geolocalización.....	69
Figura 61: Resultados posibles de SUS	76
Figura 62: Ejemplo de Trello	89

Capítulo 1: INTRODUCCIÓN A MUVI

1.1 Introducción

Cuando un grupo de amigos va al cine, casi siempre surge el problema de qué película se quiere ver y suele haber un desacuerdo porque cada uno tiene gustos diferentes. También ocurre que fuera del cine, cuando una persona ve la portada de una película que le llama la atención, quiere tener la información de dicha película para comprobar si es de su agrado.

Los cines cada semana cambian su cartelera añadiendo unas películas y quitando otras. Los usuarios se ven inmersos en una cantidad de información en constante cambio, por lo que puede ser difícil inclinarse por una película u otra.

Los usuarios quieren consultar las películas de la cartelera para, por ejemplo, leer la sinopsis, ver las valoraciones de otros usuarios, críticas de portales cinéfilos y visualizar el tráiler. Este dilema de qué película ir a ver lleva un determinado tiempo, y algunas veces, incluso no se consigue la satisfacción que se buscaba.

Por todo ello decidimos lidiar con estas dificultades de los usuarios desarrollando una aplicación móvil, que llamamos MUVI, que ofrece la posibilidad de obtener la información deseada con solo enfocar con la cámara de su móvil la portada de las diferentes películas, mostrándose la información necesaria desde una visión nueva e interactiva con el dispositivo gracias a la realidad aumentada. Gracias a MUVI se puede obtener la información relevante de una película en un simple vistazo, como el tráiler, la sinopsis, las categorías donde se enmarca, la posibilidad de valorarla y además ver las valoraciones reales o mediante predicción, en el caso de que aún no la hayan valorado, de tus amigos.

Durante los últimos años se está presentando un auge de algunas tecnologías como la realidad aumentada, los dispositivos Beacon y los sistemas de recomendación, las cuales se integrarán en nuestra aplicación para ayudarnos a solventar el problema inicial.

Aprovechando que cada vez los dispositivos tienen mejores cámaras y mayor potencia, se nos posibilita la integración de la realidad aumentada, encargada de mezclar la realidad que capta la cámara con la información que deseamos transmitir alrededor de la imagen capturada, dotándonos de una herramienta para que el usuario pueda aumentar la información sobre el entorno que le rodea.

El uso de dispositivos inteligentes de bajo consumo como los dispositivos de localización en interiores, también llamados *beacons*, nos abre un abanico de posibilidades en el desarrollo de

aplicaciones en interiores, lo que nos permiten posicionar varios objetos o personas y posibilitándonos el reconocer qué personas del grupo están en el cine en cada momento.

Para solventar el problema de los gustos de los amigos que van al cine se usarán sistemas de recomendación grupales, que consisten en la abstracción de los gustos de cada uno de los integrantes del grupo sobre una determinada película, basándose en películas relacionadas.

Cabe destacar la importancia de la sinergia que existe entre todas las tecnologías que usaremos para desarrollar nuestra aplicación, con el objetivo de ofrecerle al usuario la información que requiere de forma interactiva y novedosa.

1.2 Objetivos

El objetivo general de nuestro proyecto es desarrollar una aplicación móvil enfocada a ayudar a los usuarios en la decisión de qué película ver cuándo van al cine o estén en el propio cine. También se presenta como un objetivo fundamental el desarrollo de esta aplicación haciendo uso de tecnologías como la realidad aumentada, dispositivos Beacon y sistemas de recomendación.

A continuación, presentaremos los objetivos que se pretenden alcanzar:

- Definir y entender la tecnología de la realidad aumentada.
- Explicar y saber lo que es la tecnología que proporcionan los *beacons*.
- Investigar sobre los sistemas de recomendación.
- Diseñar la arquitectura necesaria para construir nuestra aplicación.
- Desarrollar una aplicación para dispositivos móviles donde integremos las tecnologías de la realidad aumentada, *beacons* y recomendación para su uso en los cines.
- Evaluar el uso de la aplicación y su aceptación con usuarios reales.

1.3 Organización de la memoria

Esta memoria está organizada de la siguiente manera:

- Para empezar, habrá una introducción que presente las motivaciones que nos han llevado a la realización del proyecto, así como los contenidos que se han desarrollado.
- En el apartado Estado del Arte, se hará un análisis de las tecnologías utilizadas más importantes. Se realizará un estudio sobre la realidad aumentada, los sistemas de localización internos y los sistemas de recomendación. Trataremos temas como las diferentes librerías de desarrollo y sistemas desarrollados.
- El apartado de Funcionalidad se adentrará en el funcionamiento de nuestra aplicación, donde trataremos la evolución de la misma desde unos prototipos hasta la versión actual

de la aplicación. Haremos hincapié en explicar el uso de las diferentes interfaces y servicios que utilizan.

- En el apartado Implementación, se tratará la arquitectura que utilizaremos en nuestra aplicación. Detallaremos las herramientas utilizadas, la metodología empleada y las pruebas previas realizadas.
- En el apartado Evaluación de Usuarios, se describe el plan seguido para evaluar la aplicación con diferentes usuarios, recogiendo las diferentes impresiones transmitidas por los mismos y utilizándolas para detectar y corregir algunos fallos de interacción y usabilidad.
- En el apartado Conclusiones y Trabajo Futuro, se recogerán todos los objetivos que nos propusimos, los que se han llevado a cabo y los problemas encontrados durante el desarrollo. Se expondrán las implicaciones que puede tener nuestro proyecto a largo plazo. Se detallarán los posibles cambios y nuevas funcionalidades que se podrán añadir en la aplicación en un trabajo futuro.
- En el apartado de Organización del Trabajo, se hablará del modelo de organización elegido, de las herramientas usadas para la comunicación entre los integrantes, para el desarrollo de la aplicación y para la gestión de las tareas que hemos ido realizando.
- En el apartado de Aportaciones al Proyecto, se detallarán los trabajos individuales realizados por cada uno de los integrantes del grupo y así, en mayor o menor medida, su implicación.

Capítulo 2: ESTADO DEL ARTE

En este apartado expondremos las tecnologías que más importancia tienen para nuestro proyecto: realidad aumentada, fuentes de información de películas, sistemas de recomendación y sistemas de localización mediante dispositivos Beacon. Estudiaremos los diferentes usos que se le da a estas tecnologías, detallaremos su funcionamiento y también veremos en qué aplicaciones del mercado se usan.

2.1 Realidad aumentada

La realidad aumentada (1) (en adelante RA) es el término que se utiliza para definir la visión de elementos virtuales interactivos (datos, información), en un entorno real, a través de un dispositivo. Para utilizarla en la actualidad, únicamente es necesario tener un dispositivo con una cámara y una pantalla, lo que hace de los Smartphone un dispositivo perfecto para utilizar esta tecnología.

Además de en dispositivos móviles y ordenadores, se están creando dispositivos centrados en RA, como las gafas de las empresas Google y Microsoft. Es por esto, que se dice que actualmente junto a la realidad virtual, la RA es una tecnología en auge. Con el paso del tiempo, más compañías están trabajando en incluir RA de múltiples formas, ya no sólo en aplicaciones de ocio y entretenimiento, como es el caso de este proyecto, sino también en otros ámbitos, como arqueología, medicina o uso militar.

Su funcionamiento consiste en mostrar, a través de la pantalla del dispositivo y sobre la imagen captada por la cámara, información virtual que complementan lo visto en la realidad y permite la interacción en tiempo real entre el usuario y la información. Para poder mostrarla, se utiliza el reconocimiento de marcadores como tarjetas de códigos QR, carteles de películas, etc.

En el siguiente apartado se explican los usos más importantes de esta tecnología.

2.1.1 Usos

La RA se está utilizando en múltiples ámbitos, entre ellos:

Arqueología

Puede utilizarse para “reconstruir” las edificaciones en yacimientos y lugares antiguos en los que únicamente quedan los restos de épocas pasadas. Un ejemplo de ello es la guía interactiva del Museo al Aire Libre del yacimiento de la Villa Romana de l’Albir, en Alicante (2). Gracias a ella, se puede visualizar en la pantalla del dispositivo una representación 3D del conjunto de las termas sobre el yacimiento y ayudar a mostrar el funcionamiento de las estancias de los baños.

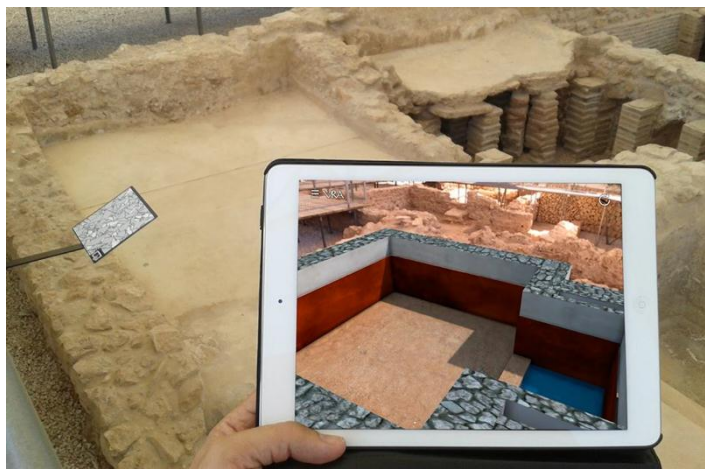


Figura 1: Ejemplo de uso en la arqueología

Arquitectura

Gracias a la RA se puede visualizar cómo quedarían construcciones en un determinado lugar gracias a la creación de objetos 3D en entornos reales, o para visualizar la maqueta de una construcción u obra a la hora de enseñar un proyecto.



Figura 2: Ejemplo de uso en la arquitectura

Medicina

Algunos de los usos de la RA en medicina son ayudar a personas autistas a interactuar en un entorno social (3), enseñanza especializada, ayuda en operaciones reconociendo tumores (4), etc.

Un ejemplo de enseñanza especializada en medicina es mARble, una aplicación desarrollada por el Colegio de Medicina de Hannover (5) para ayudar a estudiantes de medicina a ganar experiencia gracias a la RA. Proyecta “enfermedades virtuales” en sujetos sanos y muestra textos, videos y gráficos sobre la enfermedad.



Figura 3: Ejemplo de uso en la medicina

Uso militar

El ejército de Estados Unidos creó cascos “inteligentes” con un visor de RA diseñado para el campo de batalla. El sistema muestra información como la localización de los enemigos, imágenes satélites o su orden actual. El software se ejecuta en un dispositivo junto al casco. Gracias a una cámara, puede rastrear lo que el soldado está viendo y mostrar la información en el monitor (6).



Figura 4: Ejemplo de uso militar

Museos

La RA en los museos está siendo utilizada para mostrar más información e interactuar con obras y objetos. Muchos museos de todo el mundo están empezando a incorporar RA, como es el caso del Museo Británico, el Museo de Londres o el Museo de América de Madrid, con la aplicación RACMA (7), la cual ayuda a mostrar las colecciones más destacadas de las culturas expuestas gracias al poder de la RA.

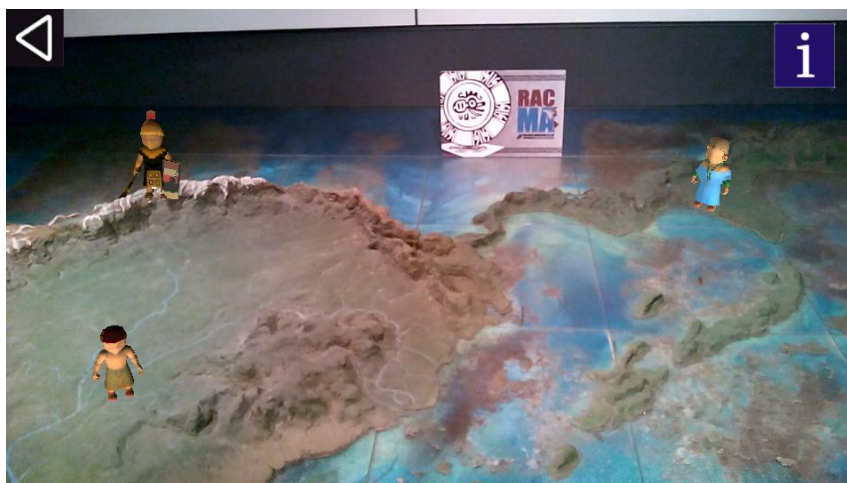


Figura 5: Ejemplo de uso en museo

Videojuegos

En los últimos años, muchas compañías han querido adentrarse en el mundo de la RA con videojuegos para consolas portátiles y *smartphones*. Quizás el máximo representante de esta categoría es la empresa española Novarama, creadores de *Invinzimals* (8), juego original de PSP, y que posteriormente salió en muchos más dispositivos. En él, con la ayuda de una tarjeta *marcadora* y la cámara de la consola o *smartphones*, asistimos a batallas de monstruos en la mesa de nuestra habitación.



Figura 6: Ejemplo de uso en los videojuegos

2.1.2 Tecnologías

Actualmente hay muchas librerías para programar y desarrollar RA. Para comprobar cuál de ellas se acercaba más a nuestras necesidades buscamos diferentes alternativas. Algunas de ellas son estas:

OpenCV

OpenCV (9) es una librería *open source* de visión artificial desarrollada por Intel. Se utiliza tanto en sistemas de seguridad con detección de movimiento como en reconocimiento de objetos o RA. Es multiplataforma, existen versiones para GNU/Linux, Mac y Windows, y tiene funciones que van desde el reconocimiento de objetos a la visión robótica.

Vuforia

Vuforia (10) es una librería que permite construir aplicaciones basadas en RA. Ofrece reconocimiento de texto, reconocimiento de imágenes, detección rápida y rastreo simultáneo y robusto de objetivos. Puede desarrollarse en Unity, Java, C++ y es compatible con iOS y Android. Su arquitectura consiste en:

- Cámara: permite captar la imagen en busca del *Target* (objetivo).
- Base de datos: puede ser local o en la nube. Almacena una colección de *Targets* para poder reconocerlos con el *Tracker*.
- Target: son los objetivos a reconocer. Pueden ser *Image Targets* (imágenes, tales fotos, páginas de revista, posters, etc.) o *Word Targets* (palabras, frases, etc.)-
- Tracker: analiza la imagen de la cámara y detecta objetos a través de la cámara con el fin de encontrar coincidencias con la base de datos.

Metaio

Compañía de RA (11) que desarrollaba software de RA comprada por Apple. Ofrece diferentes productos para crear aplicaciones de RA:

- Metaio SDK: herramienta para crear aplicaciones para Android, iOS y Windows con un plugin adicional en Unity.
- Metaio Creator: permite crear aplicaciones RA sin conocimientos especializados en programación, a través de una interfaz Drag and Drop.
- Metaio CVS: servicio de búsqueda para reconocer imágenes instantáneamente.

Mixare

Mixare (12) es una aplicación que trabaja de forma totalmente autónoma y que está disponible para el desarrollo de aplicaciones propias publicada bajo licencia GPLv3. Está disponible para iOS y Android.

ARToolKit

ARToolKit (13) es una librería software para la realización de aplicaciones de RA. Utiliza algoritmos de visión computacional para buscar el punto de vista de los usuarios. Entre sus características se encuentran la distribución completa del código fuente, calibración sencilla de la cámara y distribuciones para Linux, Mac OS, Windows OS.

Wikitude

Wikitude (14) es una tecnología para dispositivos móviles de pago que posee un kit de desarrollo de RA.

Algunas de las características de Wikitude son:

- API nativa y API JavaScript: permite la programación para Android/iOS o con tecnologías web como JavaScript.
- Servicios de localización basados en datos de geolocalización.
- Posee extensiones para Cordova, Titanium, Xamarin y Unity.
- Permite el desarrollo de aplicaciones para Smart Glasses.
- Permite añadir videos, modelos 3D, imágenes y elementos HTML.
- Incorpora reconocimiento de imágenes y seguimiento: permite reconocimiento en la nube y en local de 1000 imágenes.

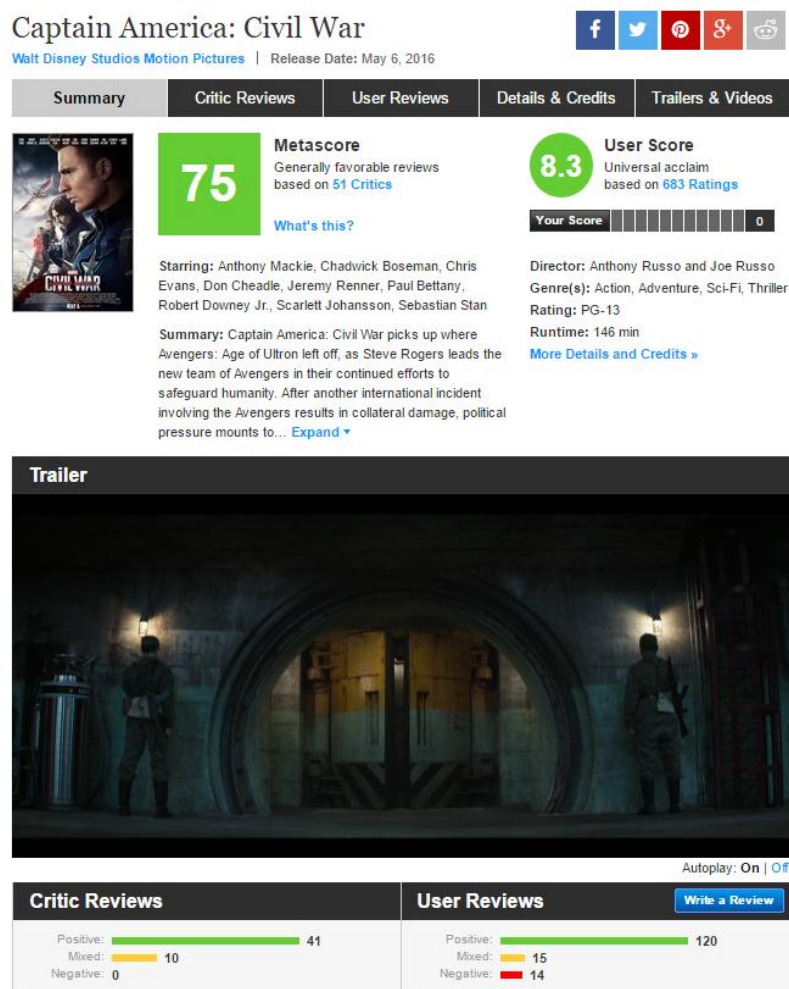
2.2 Fuentes de información de películas

Debido a que la aplicación muestra información de películas y valoraciones, y esta información es cambiante y dinámica, necesitamos saber de dónde extraer esta información. Por ello investigamos las principales webs y bases de datos de películas para comprobar cuál de ellas ofrece todo lo que queremos mostrar.

2.2.1 Metacritic

Metacritic (15) es una de las webs más conocidas de análisis de Estados Unidos, propiedad de la cadena de televisión CBS. Es conocida por agrupar todos los análisis de películas, videojuegos, series y música de la prensa especializada de todo el mundo y hacer una media con todas ellas. Además, todos los usuarios pueden valorar igualmente y realiza también una media con todas las opiniones.

Poseía una API no oficial que finalmente fue cerrada por los creadores del sitio.



Captain America: Civil War
Walt Disney Studios Motion Pictures | Release Date: May 6, 2016

Summary | Critic Reviews | User Reviews | Details & Credits | Trailers & Videos

Metascore
75
Generally favorable reviews
based on 51 Critics
[What's this?](#)

User Score
8.3
Universal acclaim
based on 683 Ratings
Your Score: 0

Starring: Anthony Mackie, Chadwick Boseman, Chris Evans, Don Cheadle, Jeremy Renner, Paul Bettany, Robert Downey Jr., Scarlett Johansson, Sebastian Stan

Director: Anthony Russo and Joe Russo

Genre(s): Action, Adventure, Sci-Fi, Thriller

Rating: PG-13

Runtime: 146 min

[More Details and Credits »](#)

Summary: Captain America: Civil War picks up where Avengers: Age of Ultron left off, as Steve Rogers leads the new team of Avengers in their continued efforts to safeguard humanity. After another international incident involving the Avengers results in collateral damage, political pressure mounts to... [Expand ▼](#)

Trailer

Autoplay: ☒ On | ☐ Off

Critic Reviews

Positive	Mixed	Negative
41	10	0

User Reviews [Write a Review](#)

Positive	Mixed	Negative
120	15	14

Figura 7: Ejemplo de Metacritic

2.2.2 IMDb

IMDb (Internet Movie Database) (16) es una de las bases de datos de cine más grandes de Internet. Contiene información de películas, actores, series de televisión, actores de doblaje e incluso de personajes ficticios. Actualmente es propiedad de Amazon.

Contiene mucha información sobre cada una de las películas: valoraciones, análisis, directores, guionistas, actores, fecha de estreno, noticias relacionadas con la película, películas parecidas, etc. Una característica poco común en las bases de datos de películas es que además de toda esta información, se incluye el elenco total de personas que han participado en la película, desde asistentes de sonido hasta extras.

OMDb (17) es la API no oficial de IMDb, gratuita, y que contiene información como el género, fecha de lanzamiento, sinopsis...

Capitán América: Civil War (2016)

Captain America: Civil War (*original title*)

PG-13 | 2h 27min | Action, Adventure, Sci-Fi | 29 April 2016 (Spain)

8,4/10 157,064 Rate This

1:01 Trailer 10 VIDEOS 482 IMAGES

Get Showtimes
In 36 theaters near [change]

Political interference in the Avengers' activities causes a rift between former allies Captain America and Iron Man.

Directors: Anthony Russo, Joe Russo

Writers: Christopher Markus (screenplay), Stephen McFeely (screenplay) | 3 more credits »

Stars: Chris Evans, Robert Downey Jr., Scarlett Johansson | See full cast & crew »

75 Metascore From metacritic.com

Reviews 732 user | 443 critic

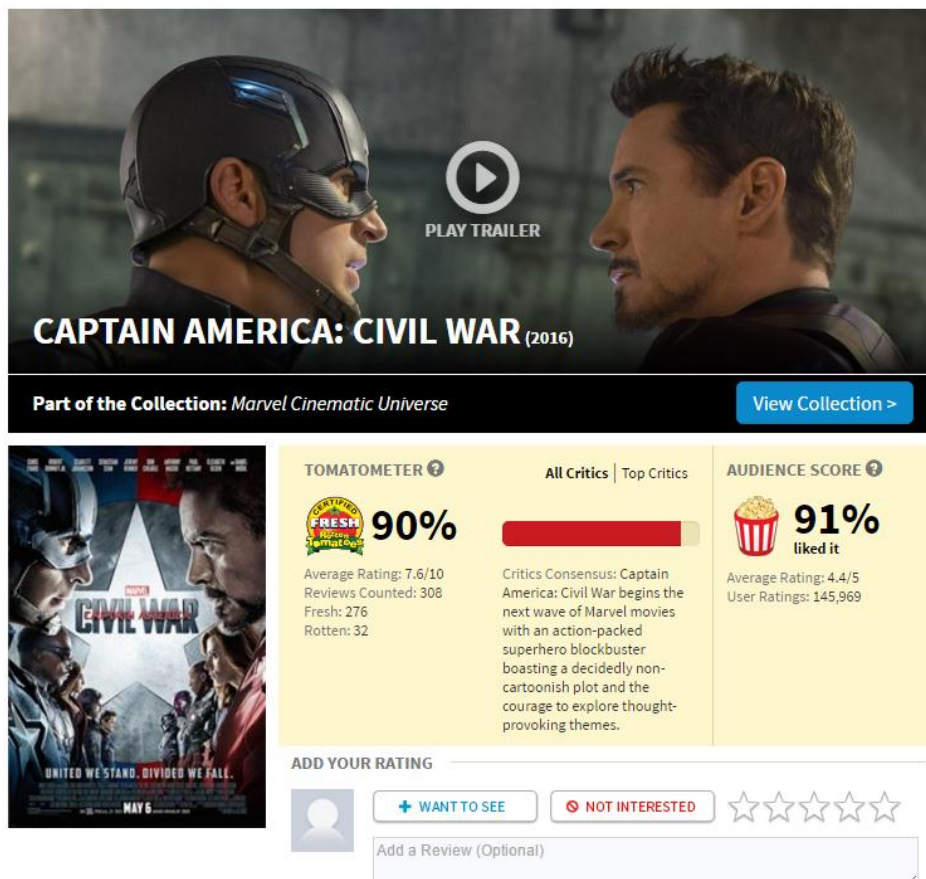
Popularity 1

Figura 8: Ejemplo de IMDb

2.2.3 Rotten Tomatoes

Rotten Tomatoes (18) es otra de las webs de opinión más importantes y conocidas de la industria del cine y la televisión. Como Metacritic, recoge críticas tanto de medios especializados como del público, además de mostrar información de la película como duración, presupuesto, casting y noticias relacionadas.

Posee una API (19) que requiere de clave de acceso y tiene un número limitado de usos por día y requiere permiso por parte del equipo de Rotten Tomatoes para poder utilizarse fuera de los Estados Unidos.



CAPTAIN AMERICA: CIVIL WAR (2016)

Part of the Collection: *Marvel Cinematic Universe* [View Collection >](#)

TOMATOMETER ? **90%** FRESH

Average Rating: 7.6/10
Reviews Counted: 308
Fresh: 276
Rotten: 32

AUDIENCE SCORE ? **91%** liked it

Average Rating: 4.4/5
User Ratings: 145,969

CRITICS CONSENSUS: Captain America: Civil War begins the next wave of Marvel movies with an action-packed superhero blockbuster boasting a decidedly non-cartoonish plot and the courage to explore thought-provoking themes.

ADD YOUR RATING

[+ WANT TO SEE](#) [NOT INTERESTED](#) ☆☆☆☆☆

Add a Review (Optional)

Figura 9: Ejemplo de Rotten Tomatoes

2.2.4 TheMovieDb.Org

TheMovieDb.org (20) es una base de datos de películas y series. Es pública, por lo que cualquier persona puede añadir una nueva película, o actualizar una existente. Entre la información que muestra se encuentra la habitual de este tipo de páginas, duración, sinopsis, valoración de los usuarios de la web, etc. Además de su presupuesto, se pueden ver los ingresos que ha obtenido la película, y como ocurre en IMDb, también se puede ver todo el reparto de la película, incluyendo operadores de cámara, maquilladores, etc.

Incluye una API (21) que también necesita una clave de acceso, pero puede usarse fuera de los Estados Unidos y muestra mucha información de la película como tráiler, sinopsis...

The screenshot displays the Rotten Tomatoes page for the movie "Capitán América 3: Civil War" (2016). The page layout includes a movie poster on the left, a main title section with a rating of 6.9/10 (1,698 votes), and several informational sections. The "Overview" section provides a synopsis of the film. The "Crew" section lists the directors (Anthony Russo, Joe Russo) and writers (Christopher Markus, Stephen McFeely). The "Cast" section features headshots and names of the main actors, including Chris Evans as Steve Rogers, Robert Downey Jr. as Tony Stark, Scarlett Johansson as Natasha Romanoff, and Sebastian Stan as James Buchanan "Bucky" Barnes. The "Backdrops" section shows a horizontal scroll of movie stills. The left sidebar contains additional details such as the original title, movie facts, and release information.

Posters Show All

Capitán América 3: Civil War (2016)

☆☆☆☆☆ 6.9/10 (1,698 votes)

Favorite Watchlist Videos (4) Lists (130) Changes Share Report

Overview

"Captain America: Civil War" continúa la historia de "Avengers: Age of Ultron", con Steve Rogers liderando un nuevo equipo de Vengadores en su esfuerzo por proteger a la humanidad. Tras otro incidente internacional relacionado con los Vengadores que ocasiona daños colaterales, la presión política fuerza a crear un sistema de registro y un cuerpo gubernamental para determinar cuándo se requiere los servicios del equipo. El nuevo status quo divide a los Vengadores mientras intentan salvar al mundo de un nuevo y perverso villano."

Tagline

No tagline has been added.

Crew

Directors: [Anthony Russo](#), [Joe Russo](#)

Writers: [Christopher Markus](#), [Stephen McFeely](#)

Show All

Cast

[Chris Evans](#) as Steve Rogers / Captai...

[Robert Downey Jr.](#) as Tony Stark / Iron Man...

[Scarlett Johansson](#) as Natasha Romanoff / BI...

[Sebastian Stan](#) as James Buchanan 'Bucky'...

Show All

Backdrops Show All

Original Title

Captain America: Civil War

Movie Facts

Part of the: [Capitán América - La Colección](#)

Status: Released

Runtime: 147

Budget: \$250,000,000

Revenue: \$206,000,000

Languages: ro, en, de, ru

Webpage: -

Release Info

2016-04-29

Figura 10: Ejemplo de Rotten Tomatoes

2.3 Sistemas de recomendación

Debido al incesante crecimiento de Internet y el exceso de información que esto conlleva, seleccionar aquello que los usuarios necesitan dentro de una cantidad de datos masiva resulta una tarea difícil y poco precisa. En consecuencia, nacen los sistemas de recomendación, los cuales facilitan la selección de información de manera rápida.

Un sistema de recomendación es un sistema inteligente que proporciona a los usuarios una serie de sugerencias personalizadas (recomendaciones) sobre un determinado tipo de elementos (ítems) (22).

A través de un sistema de recomendación se pueden hacer sugerencias de diversas naturalezas: desde cualquier tipo de producto para comprar, escuchar o consumir, hasta usuarios con intereses similares.

2.3.1 Tipos

Los sistemas de recomendación habitualmente suelen basarse en algoritmos de filtrado colaborativo. Estos algoritmos pueden ser de dos tipos: basados en el usuario o basados en el contenido.

Filtrado colaborativo basado en el usuario

Los filtrados basados en el usuario tienen como base de su predicción las valoraciones realizadas por un usuario. El sistema analiza el histórico de compras o valoraciones que ha dado a productos, y busca otros usuarios que hayan tomado decisiones parecidas, es decir, que se parezcan a él.

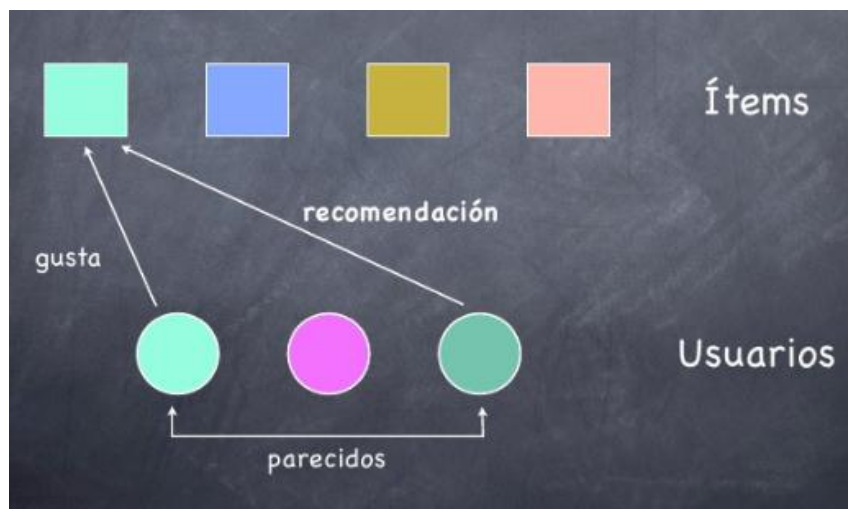


Figura 11: Filtrado colaborativo basado en el usuario

Filtrado colaborativo basado en el producto

Los filtrados basados en el producto tienen como base de su predicción las valoraciones recibidas de los productos. El proceso es similar al del filtrado basado en el usuario, con la diferencia de que en este caso se buscan similitudes entre los productos valorados en lugar de entre los usuarios.

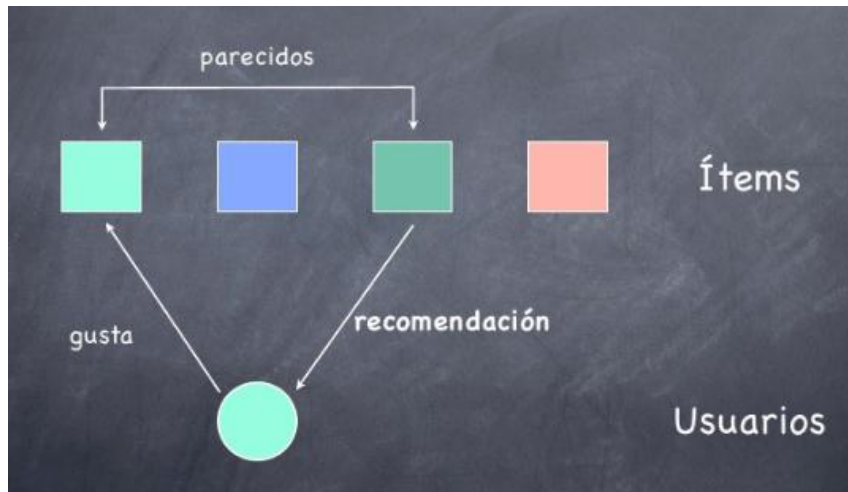


Figura 12: Filtrado colaborativo basado en el contenido

Otras técnicas utilizadas para los sistemas de recomendación son:

Recomendación basada en contenido

La recomendación basada en contenido consiste en recomendar productos a partir de las características del producto y del perfil de los intereses del usuario, seleccionando los productos que tienen un alto grado de proximidad a las preferencias de dicho usuario. A diferencia de los sistemas basados en filtrado colaborativo, puede proporcionar recomendaciones de nuevos productos, aunque no haya valoraciones disponibles.

Recomendación basada en conocimiento

La recomendación basada en conocimiento consiste en recomendar productos en función de una serie de requisitos especificados por el usuario, ya sean restricciones que el producto ha de cumplir (*constraint-based*) o características ideales que el producto ha de tener (*case-based*). En el caso de no encontrarse una posible recomendación en base a estos requisitos, el usuario debe “modificar” los mismos.

Recomendación híbrida

Consiste en combinar las técnicas explicadas anteriormente (dos o más), tanto las de filtrado colaborativo, como las basadas en contenido y en conocimiento, con el objetivo de suplir las carencias que tienen cada una de ellas.

2.3.2 Tecnologías

Algunas de las librerías disponibles para la construcción de sistemas de recomendación son las siguientes:

LibRec

LibRec (23) es una librería Java de licencia GPL para sistemas de recomendación que implementa una serie de algoritmos de recomendación. Se compone de tres elementos principales: interfaces, estructuras de datos y algoritmos de recomendación.

LensKit

LensKit (24) es una herramienta de recomendación basada en Java de GroupLens. Proporciona una API común para algoritmos de recomendación, un *framework* para la evaluación offline del rendimiento de recomendación e implementaciones modulares de algoritmos de filtrado colaborativo comunes.

Mahout

Mahout (25) es un proyecto de Apache cuyo objetivo es construir un entorno para la creación rápida de aplicaciones escalables de aprendizaje automático. Dispone de una librería Java con algoritmos de distintas áreas: de agrupamiento, de patrones de minería, de clasificación, de regresión, evolutivos y de recomendación.

2.4 Sistemas de localización con dispositivos Beacon

Estos sistemas de localización se asemejan a un faro que transmite repetidas veces una señal que otros dispositivos pueden visualizar. En lugar de emitir señales de luz, estos transmiten señales de radio en el espectro de frecuencias captadas por el Bluetooth. Por lo tanto, cualquier Smartphone con dispositivo Bluetooth podrá captar esta señal siempre que esté dentro del rango de emisión del Beacon.

Estos dispositivos permiten ayudar a desarrollar aplicaciones pensadas para su uso dentro de grandes edificios, donde los sistemas de localización convencionales no disponen de un soporte válido para dicha función.

2.4.1 Usos

El uso de localización mediante dispositivos Beacon se puede dar en múltiples ámbitos (26):

Centros comerciales

Los centros comerciales cada vez son más grandes y los usuarios desconocen en gran medida la localización de las diferentes tiendas. Es por ello que mediante la implantación de estos dispositivos se puede dirigir a las personas hacia la tienda que quiera. Además, estos dispositivos nos permitirán la interacción con nuestros dispositivos móviles y los artículos de la tienda con simplemente acercándonos a un, pudiendo así realizar varias tareas como consultar el precio, características e incluso comprar el artículo en cuestión.



Figura 13: Funcionalidad de los dispositivos Beacon dentro de un centro comercial

Seguridad

Se puede implementar también para un tratamiento de accesos en un edificio. Con el uso de estos dispositivos en todo momento las personas estarán localizadas dentro del edificio y permitiendo el acceso a aquellas personas que tengan permisos.

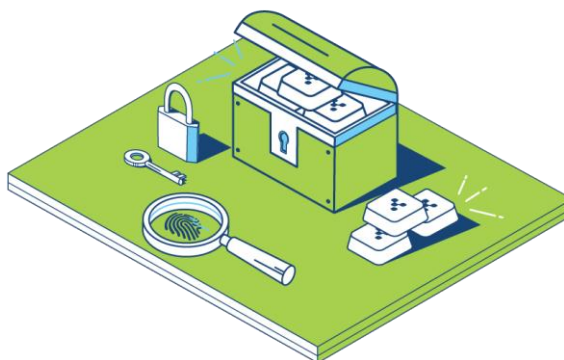


Figura 14: Ilustración de Seguridad

Recabar información sobre zonas de tránsito

Las balizas también son muy útiles a la hora de recopilar información para saber cuáles son las zonas de más tránsito en una tienda o conocer las áreas más concurridas de un centro comercial, un aeropuerto o un congreso. También es un buen indicador del éxito o fracaso de una promoción o un nuevo producto lanzado recientemente al mercado, incluso para medir el pulso de un evento y mejorar la experiencia futura.

2.4.2 Tecnologías

Estimote Beacons

Estimote Beacons (27), son Beacons y Pegatinas sin necesidad de conectar a una fuente de carga, con pequeños sensores inalámbricos que se pueden adjuntar cualquier lugar u objeto. Con el SDK Estimote se puede reconocer lugares y objetos estrechamente relacionados con los diferentes Beacons. Las plataformas en las que son usadas son iOS con iBeacon y Android con Eddystone siendo esta última de software libre.

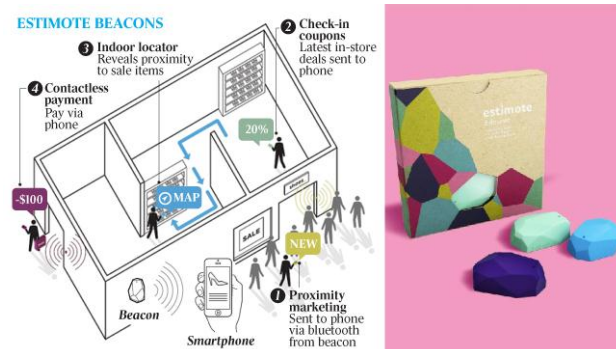


Figura 15: Ilustración y aplicación de los Estimote Beacons

RadBeacon

RadBeacon (28), ofrece hasta tres tipos de Beacons dependiendo de las necesidades de conectarlo con una fuente de carga para su funcionamiento yendo desde conexión USB, a toma eléctrica o de forma autónoma. Estos dispositivos están más pensados para dar una gestión y seguridad de estas balizas mucho más avanzada, pudiendo configurar y supervisar las diferentes balizas mediante un servicio en la nube y siendo compatible estos dispositivos con servicios API REST. Son compatibles con iBeacon(iOS), AltBeacon y Eddystone Beacon(Android).



Figura 16: Ilustración de los diferentes dispositivos de RadBeacon

Bluecat Beacon

Bluecat Beacon (29) utilizan como forma de fuente de carga pilas convencionales, donde nos ofrece unas herramientas para monitorizar y configurar nuestros Beacons en la nube. Ofrece un SDK bastante claro para que en unos minutos podamos realizar nuestro primer proyecto. Es compatible con iBeacon(iOS) y Eddystone(Android).



Figura 17: Partes de que se compone Bluecat Beacon

Kontakt.io Beacon

Kontakt.io (30) nos ofrece un único formato de Beacon el cual no necesita ninguna fuente externa de recarga. Además, nos pone a nuestra disposición las siguientes herramientas:

Proximity Web Panel: Es una plataforma fácil de usar que permite configurar y administrar la infraestructura de los Beacons, donde permite organizar despliegues en lugares virtuales, cambiar la configuración y los perfiles de las balizas de forma individual o en masa.

Proximity REST API: Permite una creación de una herramienta de configuración y administración o usar la plataforma ya existente que ofrece para los desarrolladores. Integra también soluciones para la seguridad administración masiva, análisis e información instantánea acerca de los Beacons.

El SDK de Kontakt.io es compatible con Eddystone (Android superiores a 4.3 para un correcto funcionamiento) y iBeacon (iOS 8.0 o superiores).



Figura 18: Ilustración del modelo Kontakt.io

Capítulo 3: FUNCIONALIDAD

En este capítulo, describiremos de todas las funcionalidades que tiene la aplicación a modo de manual de usuario, viendo a través de diagramas cuál sería el flujo de estados de la aplicación.

MUVI es una aplicación, como ya hemos comentado anteriormente, basada en RA, con la que se puede obtener información sobre películas utilizando las carteleras de las películas. Además, se puede compartir dicha información, almacenar tus películas favoritas y valorarlas. Debido a que MUVI está pensada para ser una aplicación social es necesario almacenar la información de los usuarios y, por eso, se requiere que todos los usuarios se registren en la aplicación antes de poder usar todas las funcionalidades.

Además, hacemos uso de los dispositivos Beacon para comprobar qué amigos están cerca de nosotros cuando escaneamos una película y así poder recomendar dicha película al grupo o no, además de poder mostrar la recomendación individual del usuario.

3.1 Funciones de la Aplicación

Antes de entrar en detalle en el flujo de la aplicación y explicar detalladamente cada interfaz que tiene nuestra aplicación, contaremos cuáles son sus funcionalidades básicas.

La principal funcionalidad de nuestra aplicación es la RA. Ahí, el usuario puede realizar la mayoría de funcionalidades de la aplicación simplemente apuntando al cartel de una película. Puede ver información sobre una película, valorar una película, ver el tráiler, compartir en Twitter, añadir a la lista de Me Gusta y ver la puntuación que la aplicación estima de la película escaneada para tus amigos conectados al mismo dispositivo Beacon que tú.

Fuera de la RA, podemos ver en una interfaz sencilla e intuitiva las películas que hemos añadido a nuestra lista de Me Gusta y la información de esas películas sin tener que volver a entrar a la RA. Además, el usuario, gracias al uso del GPS, puede ver las películas que sus amigos cercanos han añadido a Me Gusta para hacerse así una idea de qué película ver.

3.2 Flujo de la Aplicación

La primera pantalla que verá el usuario es la interfaz de Registro. Aquí el usuario introducirá sus datos para registrarse. Una vez registrados, lo siguiente es iniciar sesión. Para ello, se tiene que introducir el nombre de usuario y contraseña con la que se haya registrado el usuario en la interfaz de Inicio de Sesión.

Una vez terminado el proceso de registro y/o inicio de sesión se accede a la interfaz de Inicio, desde la cual se puede acceder a todas las funcionalidades de la aplicación. Para ello, se

dispone de una barra de menú desplegable a la que se puede acceder pulsando sobre el botón de la esquina superior izquierda o simplemente deslizando el dedo de izquierda a derecha. Además del menú se puede acceder a la RA pulsando sobre la imagen principal que hay en la pantalla de inicio.

Cuando el usuario entra en la interfaz de RA se abrirá la cámara del dispositivo móvil que se esté utilizando. Para poder capturar los posters de películas lo único que hay que hacer es enfocar con la cámara sobre el poster. Si la aplicación detecta el poster, aparecerán varios botones alrededor de él. Los botones tienen diversas funciones: añadir la película a la lista de Me Gusta, valorar la película, ver tráiler, etc. Todas estas funcionalidades se explicarán en detalle más adelante.

De vuelta a la interfaz de Inicio, el usuario puede elegir entre varias opciones. En la 3.4.5 Interfaz de Me Gusta puede ver todas las películas que haya guardado desde la 3.4.4 Interfaz de Realidad Aumentada. Estas películas son interactivas, de forma que si el usuario pulsa sobre la imagen de la película, la aplicación le redirigirá a la 3.4.6 Interfaz de Ficha de Película. En esta interfaz se muestra toda la información relevante de la película: título, sinopsis, género, etc.

En la siguiente figura se muestra el flujo de todas las interfaces de la aplicación, desde la interfaz de Registro hasta la de Ficha de Película.

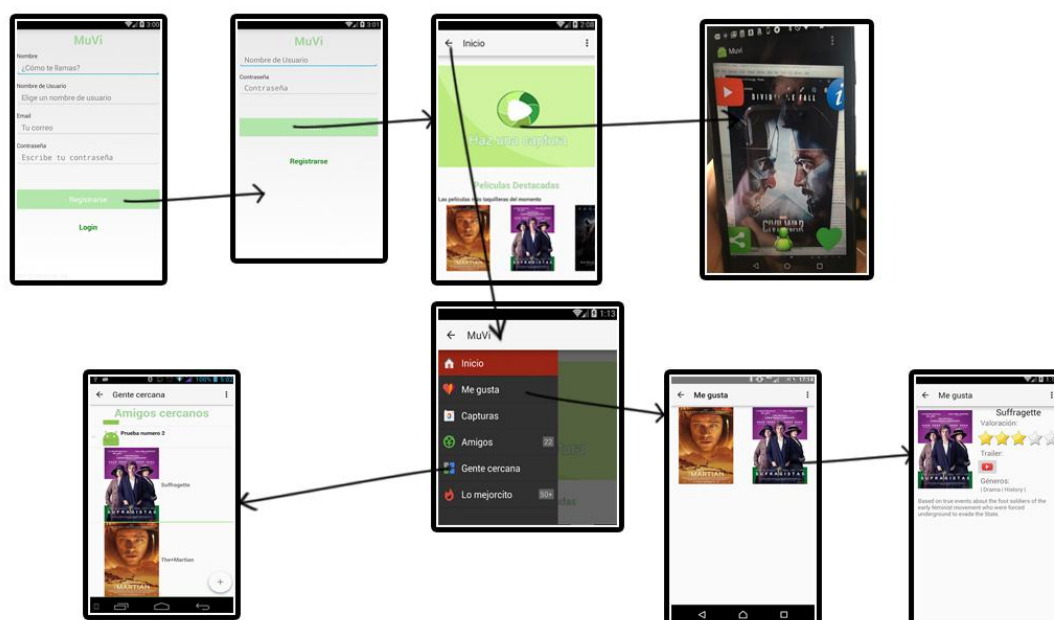


Figura 19: Flujo de las interfaces de la aplicación

3.3 Primeros prototipos

La aplicación que en un principio teníamos pensada y la que finalmente ha resultado no se parece mucho. En un principio, se pensó en hacer secciones diferentes como la búsqueda de películas, así como un historial de todas las capturas de la RA. Estas primeras ideas se plasmaron en unos dibujos sencillos como primeros prototipos.



Figura 20: Prototipo de interfaz de búsqueda



Figura 21: Ilustración del modelo Kontakt.io

Según avanzábamos en la definición de la aplicación, la mayoría de las posibles funcionalidades se desechaban, ya que no se centraban en lo que de verdad importaba, la RA. Para los siguientes prototipos utilizamos herramientas específicas como proto.io. De este modo, nos acercábamos a una primera versión de las interfaces.

Para la interfaz de inicio se pensó en tener la mitad de la pantalla con una pre-visualización de la cámara, para que así el usuario pudiera capturar posters nada más entrar en la aplicación. Sin embargo, se rechazó esta idea porque tener la cámara encendida siempre que se esté usando la aplicación, reduciría en exceso la memoria y usaría demasiado la batería del dispositivo.

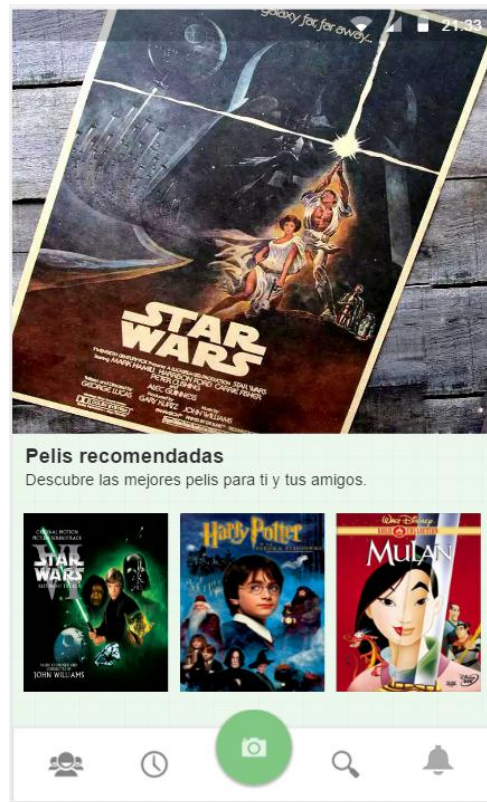


Figura 22: Primer prototipo de la interfaz de inicio

Como se puede observar en la figura, este prototipo constaba de una barra de herramientas en la parte de abajo. Esta idea fue perdiendo peso debido a que si se añadieran más funcionalidades tendríamos una barra demasiado llena de iconos. Para arreglar este problema nos decantamos por usar un menú desplegable.

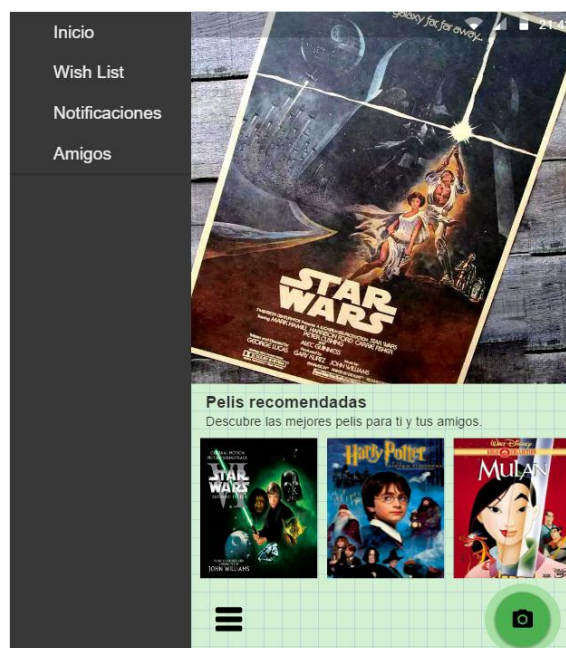
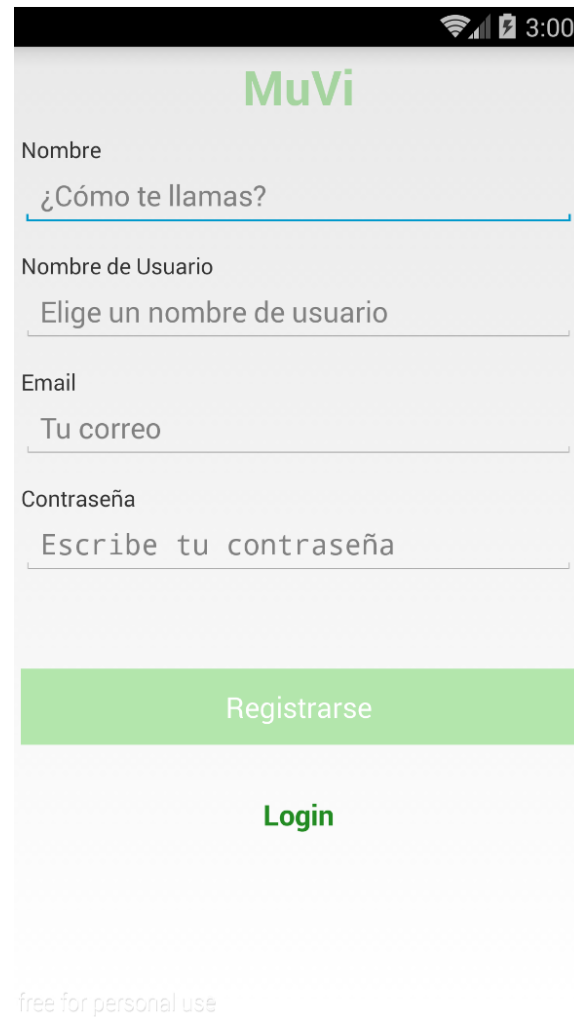


Figura 23: Segundo prototipo de la interfaz de inicio

3.4 Interfaz de usuario

En los próximos apartados se explican las diferentes interfaces de usuario de la aplicación:

3.4.1 Interfaz de Registro



MuVi

Nombre
¿Cómo te llamas?

Nombre de Usuario
Elige un nombre de usuario

Email
Tu correo

Contraseña
Escribe tu contraseña

Registrarse

Login

free for personal use

Figura 24: Interfaz de registro

El objetivo de esta interfaz es que los usuarios se den de alta en la aplicación. Se trata de un registro básico en el que se requiere que el usuario escriba su nombre, un nombre de usuario, su correo y una contraseña. Como hemos explicado anteriormente, al ser una aplicación social es necesario guardar la información de los usuarios, por eso es necesario que todos los usuarios se registren.

La aplicación hace varias comprobaciones de los datos insertados: si el nombre de usuario es único, si ningún campo está vacío y si el correo tiene un formato válido.

Si alguna de estas condiciones no se cumple, la aplicación mostrará un mensaje de error indicando que es lo que el usuario tiene que hacer. Si el registro es exitoso, la aplicación redirigirá a la interfaz de “Iniciar Sesión” para que el usuario pueda empezar a utilizar los servicios. Cuando el usuario pulse “Login” se abrirá la interfaz de “Iniciar Sesión”.

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad:

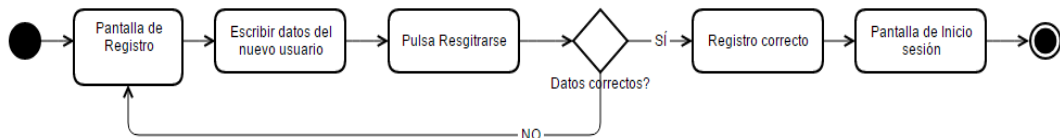


Figura 25: Diagrama de actividad de registro

3.4.2 Interfaz de Iniciar Sesión

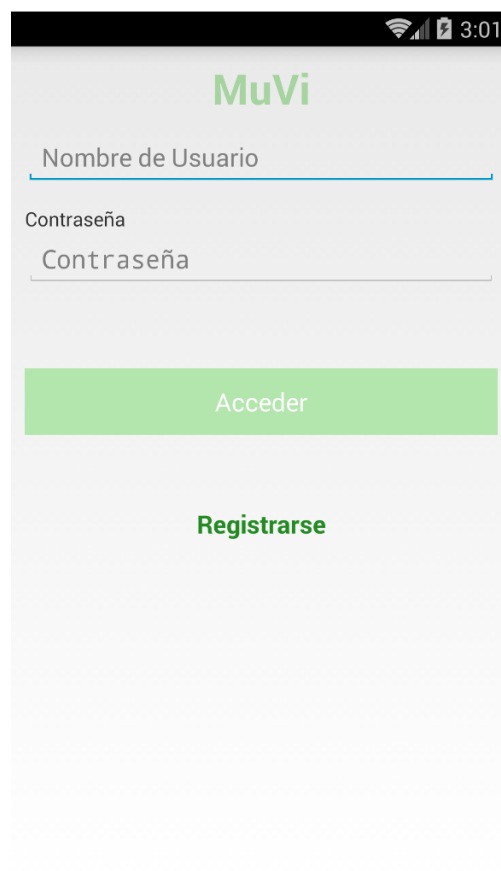


Figura 26: Interfaz de inicio de sesión

El objetivo de esta interfaz es iniciar sesión en la aplicación para poder comenzar a hacer uso del resto de funcionalidades. Para ello, el usuario debe de introducir el nombre de usuario y contraseña que utilizó cuando hizo el registro. Cuando el usuario pulse “Acceder” se comprueba que los datos introducidos son correctos y que éste existe en la base de datos de la aplicación. Hay dos opciones: si existe y los datos son correctos: entonces se inicia la sesión y se abre la ventana de inicio. Si no existe o los datos son incorrectos, entonces se notifica al usuario para que corrija los datos y vuelva a intentarlo. Cuando el usuario pulse “Registrarse” se abrirá la interfaz de “Registro”.

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad:

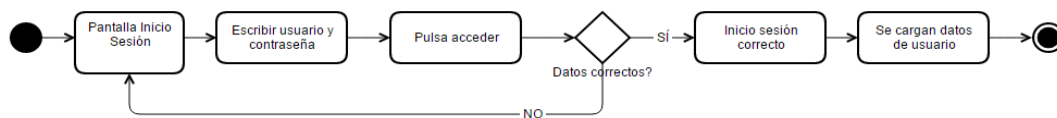


Figura 27: Diagrama de actividad de la interfaz de inicio de sesión

3.4.3 Interfaz de Inicio

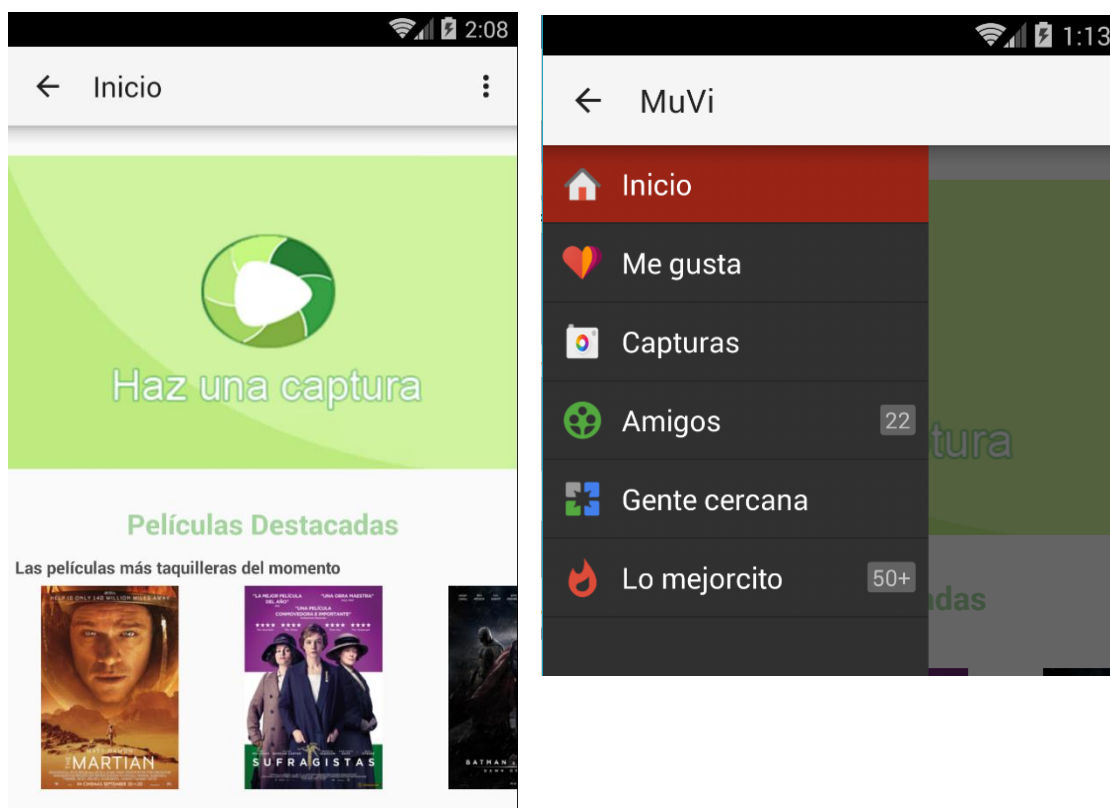


Figura 28: Interfaz de inicio y menú lateral

La interfaz de Inicio es la interfaz principal de la aplicación. Desde ella se puede acceder a cada una de las funcionalidades de la aplicación. Consta de dos elementos:

1. Una imagen principal “pulsable” con el que el usuario puede ir a la interfaz de RA.
2. Un carrusel con las películas que están arrasando en taquilla.

Además, esta interfaz tiene una barra de navegación en la parte superior con la que se puede acceder al menú. El menú tiene una lista de botones para poder ir a cualquier punto de la aplicación.

La finalidad de esta interfaz es dar la bienvenida a los usuarios. Una vez realicen correctamente el registro y posterior inicio de sesión, serán la primera pantalla que vean.

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad:

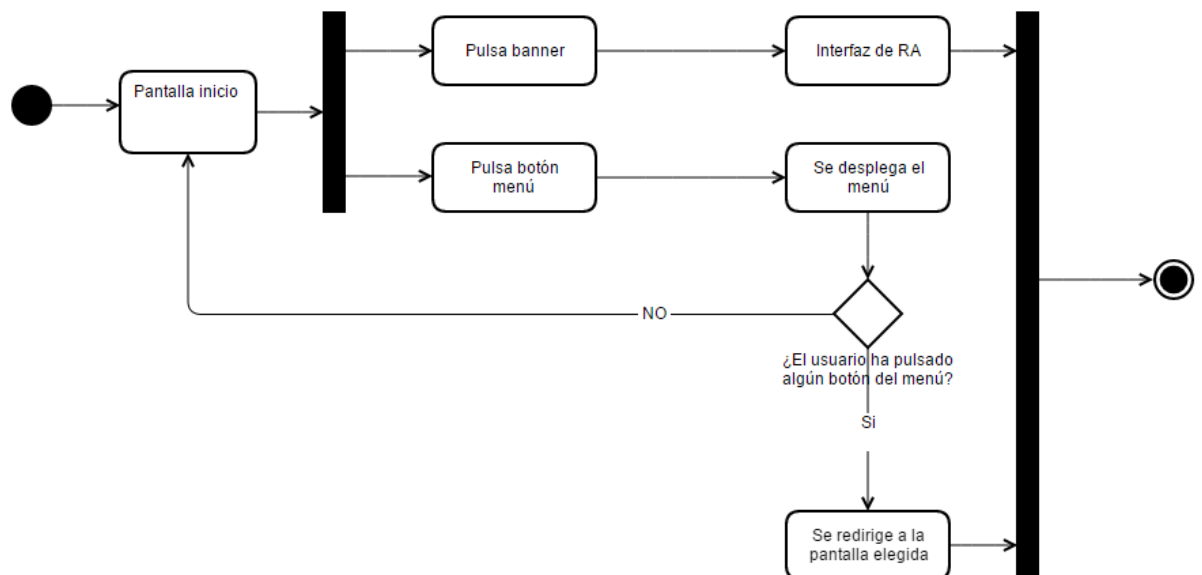


Figura 29: Diagrama de actividad de inicio

3.4.4 Interfaz de Realidad Aumentada

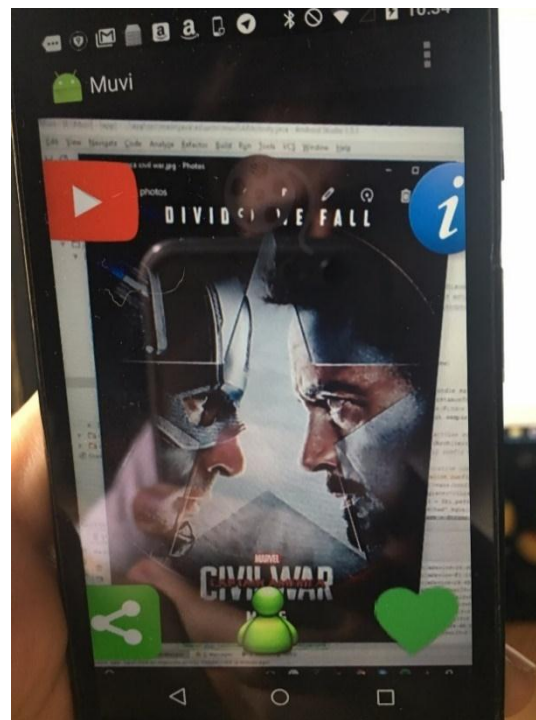


Figura 30: Interfaz de RA

La interfaz de RA es el eje de la aplicación. Desde ella se puede ver la información de la película a la que se está apuntando. Para ello, únicamente es necesario apuntar con la cámara del dispositivo móvil al cartel de la película para que la aplicación la reconozca. Una vez reconocida, aparecen diferentes iconos, cada uno con una funcionalidad específica. Para ejecutar la funcionalidad de cada uno solo es necesario pulsar sobre el botón de la RA que en concreto. Éstas funcionalidades son las siguientes:

- Ver el tráiler de la película en la misma RA: es necesario pulsar el botón de reproducción que se encuentra arriba a la izquierda. Se puede pausar pulsando sobre él, y si se pierde de vista el cartel y se vuelve a apuntar a él, el tráiler se reproduce desde el punto en el que se quedó.



Figura 31: Trailer en la interfaz de RA

- Valorar una película (siendo 5 la máxima puntuación y 1 la mínima): es necesario pulsar el botón de valoración que se encuentra arriba en el centro. Si no se ha valorado anteriormente la película, aparecerán unas estrellas huecas, pulsando sobre la estrella que quieras le darás una valoración. Una vez valorada una película, no puede volver a valorarse.



Figura 32: Valoración en RA

- Ver la información de la película: es necesario pulsar el botón de información que se encuentra arriba a la derecha. La información disponible es el nombre completo, la descripción de la película, la valoración de IMDb y los géneros de la película.



Figura 33: Información de la película en RA

- Añadir la película a favoritos (incluyéndola en la lista de Me Gusta): es necesario pulsar sobre el botón de favoritos que se encuentra abajo a la derecha. Una vez pulsado, el botón cambiará de color a rojo, indicando así que la película ha sido añadida con éxito.

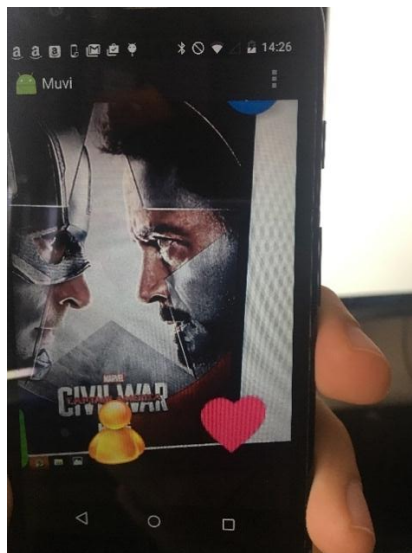


Figura 34: Información de la película en RA

- Compartir en Twitter: es necesario pulsar sobre el botón de abajo a la izquierda, y después pulsar sobre el botón de Twitter que aparecerá justo encima del anterior. Una

vez pulsado, el usuario saldrá de la aplicación para redactar el *tweet* en la aplicación de Twitter o en el navegador.



Figura 35: Información de la película en RA

- Ver la valoración que tu grupo de amigos le ha dado a la película: es necesario pulsar el botón que se encuentra abajo en el centro, el cual cambia de color dependiendo de la recomendación de la película para el grupo (rojo-mala, amarillo-normal, verde-buena). Si un amigo no ha valorado la película, se le asigna una valoración mediante el sistema de recomendación. Se muestran las valoraciones individuales mediante caras y no hay distinción entre valoración dada y valoración recomendada. Para realizar este paso, la aplicación hace uso de los dispositivos Beacon y el *bluetooth*. Nada más entrar en la RA, se busca mediante *bluetooth* algún dispositivo Beacon cercano y se conecta a él. Una vez en él, la aplicación busca a los otros usuarios conectados a ese mismo dispositivo Beacon para mostrar a los amigos.

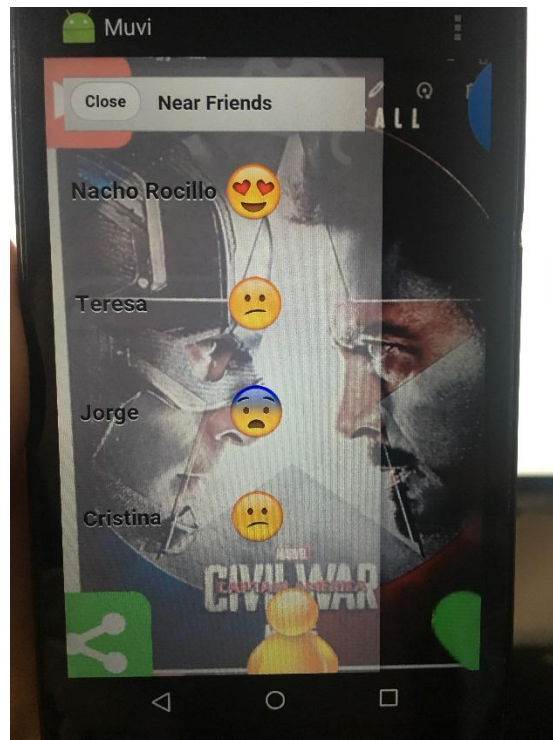


Figura 36: Amigos cercanos en RA

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad:

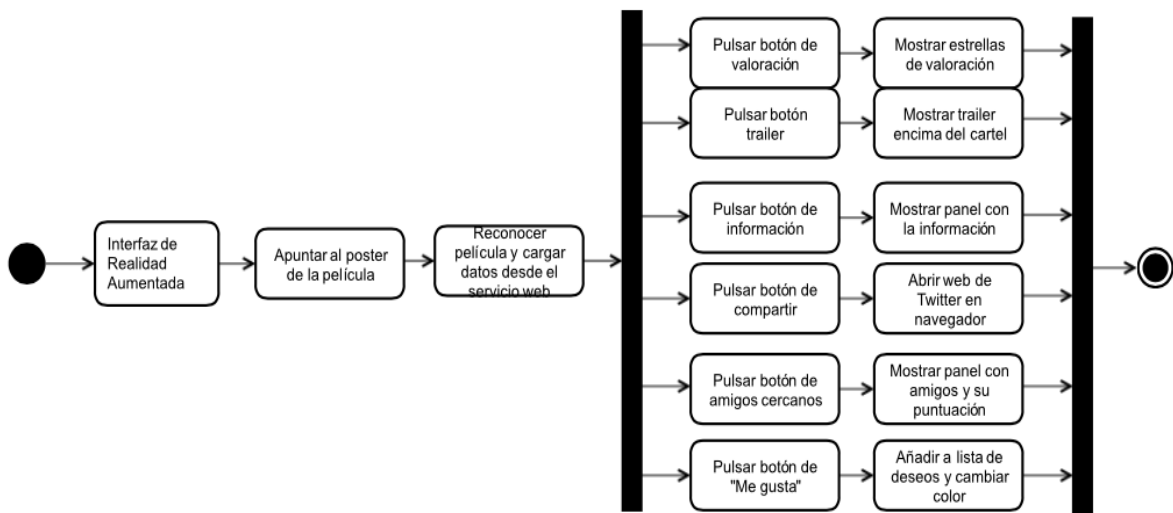


Figura 37: Diagrama de actividad de la RA

3.4.5 Interfaz de Me Gusta

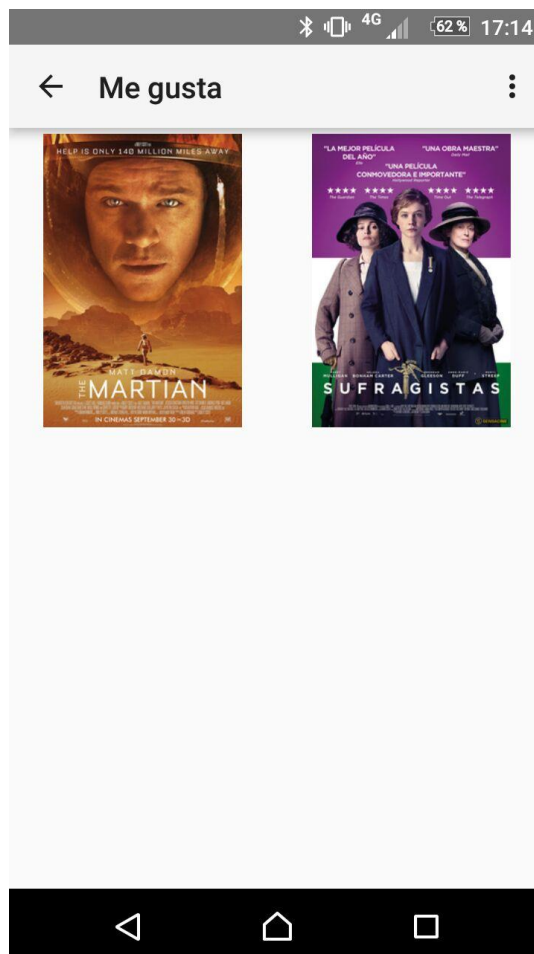


Figura 38: Interfaz de Me Gusta

El objetivo de esta interfaz es mostrar al usuario todas las películas que haya añadido a favoritos en la interfaz de RA. Se mostrarán los carteles de cada película en formato *grid*.

Estos carteles son interactivos, es decir, si el usuario pulsa en uno de ellos, la aplicación le redirigirá a la 3.4.6 Interfaz de Ficha de Película

Esta interfaz está diseñada a modo de recordatorio, para que los usuarios puedan ver las películas que le gustan y su información en cualquier momento.

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad.

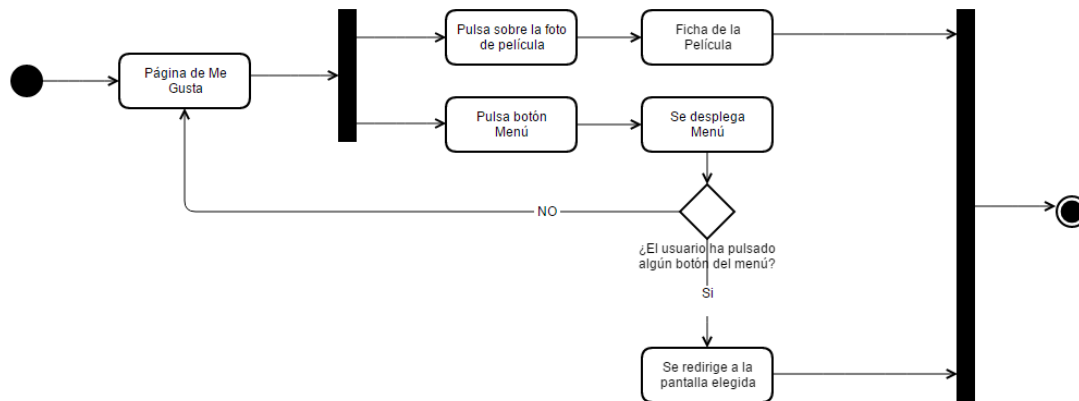


Figura 39: Diagrama de actividad de Me Gusta

3.4.6 Interfaz de Ficha de Película

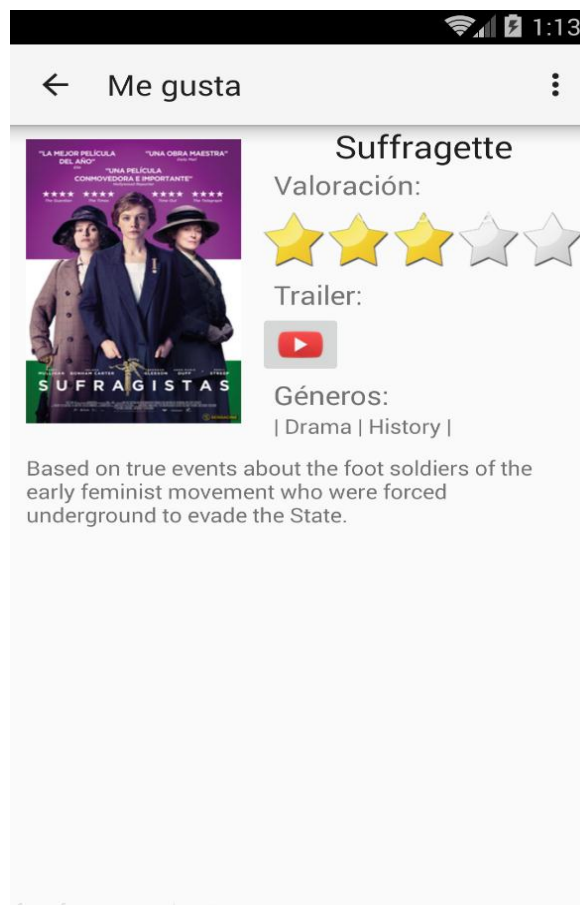


Figura 40: Interfaz de ficha de película

En esta interfaz se ofrece una descripción de una película. Para ello, primero se tiene que seleccionar dicha película desde la 3.4.5 Interfaz de Me Gusta . Se muestran diferentes datos de la película, como son:

- Título: se muestra el Título en el idioma original de la película.
- Foto del cartel: se muestra la foto del cartel de la película.
- Valoración: se muestran cinco estrellas, que dependiendo de la valoración que tenga la película, serán amarillas o grises. Por ejemplo, la película “Marte” tiene una valoración de 3 sobre 5, entonces esta película tendrá 3 estrellas amarillas y 2 grises. Las valoraciones que se muestran son las que haya en la fuente externa. En este caso, las valoraciones de IMDb.
- Tráiler: es un botón que al pulsarlo redirige al tráiler en YouTube.
- Géneros: se muestran los géneros a los que pertenece la película.
- Sinopsis: se muestra un resumen de la trama de la película. El texto está en el idioma original.

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad.

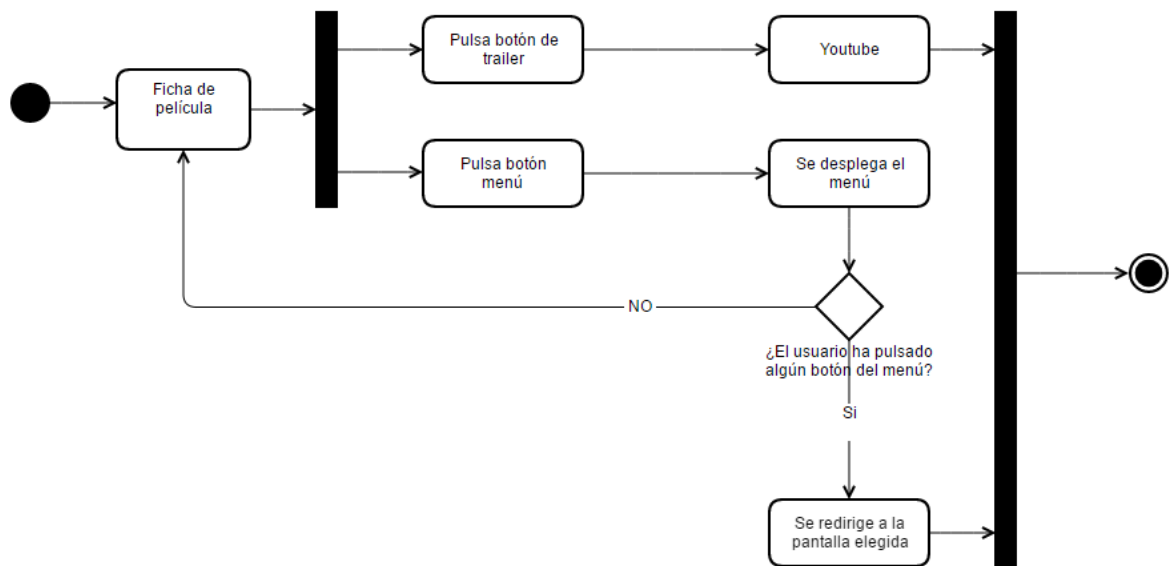


Figura 41: Diagrama de actividad de Ficha de Película

3.4.7 Interfaz de Gente Cercana

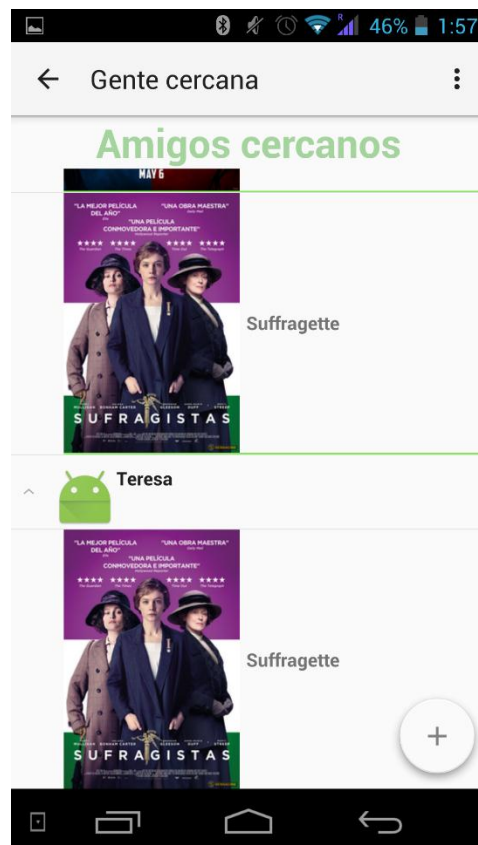


Figura 42: Interfaz de Gente Cercana

La finalidad de esta interfaz es ofrecer al usuario una idea de que películas están teniendo más éxito entre las personas que le rodean, ofreciéndole una idea de qué película podría ver.

Esta interfaz se dispone a mostrar las personas cercanas al usuario, pudiendo consultar las películas favoritas de cada una que se haya añadido desde la RA.

Primero, se mostrará una lista de personas y si el usuario quiere saber los gustos de las mismas tendrá que pulsar sobre una de ellas. Por otro lado, si el usuario vuelve a accionar este botón se ocultará la lista de películas de dicha persona. Por último, el usuario puede ir a la RA pulsando el botón flotante de la esquina derecha.

A continuación, se presenta un diagrama de actividad para entender el flujo de actividad que tiene esta funcionalidad.

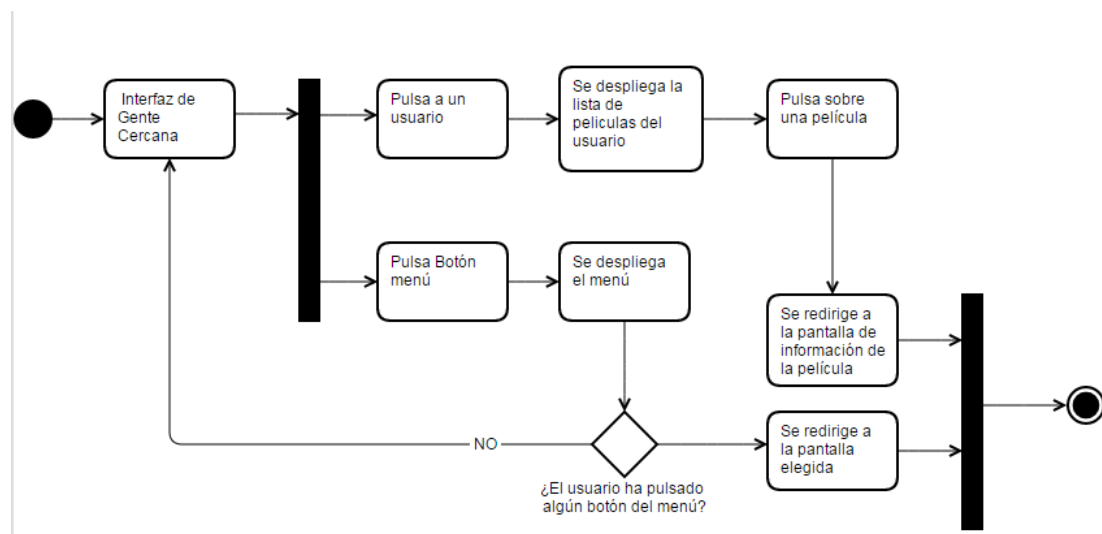


Figura 43: Diagrama de actividad de Gente Cercana

Capítulo 4: IMPLEMENTACIÓN

En este apartado se detalla la arquitectura de la aplicación y el proceso de creación de la misma, incluyendo las distintas tecnologías implicadas, los lenguajes de programación y entornos de desarrollo utilizados, así como las diferentes pruebas realizadas como toma de contacto con dichas tecnologías.

4.1 Arquitectura

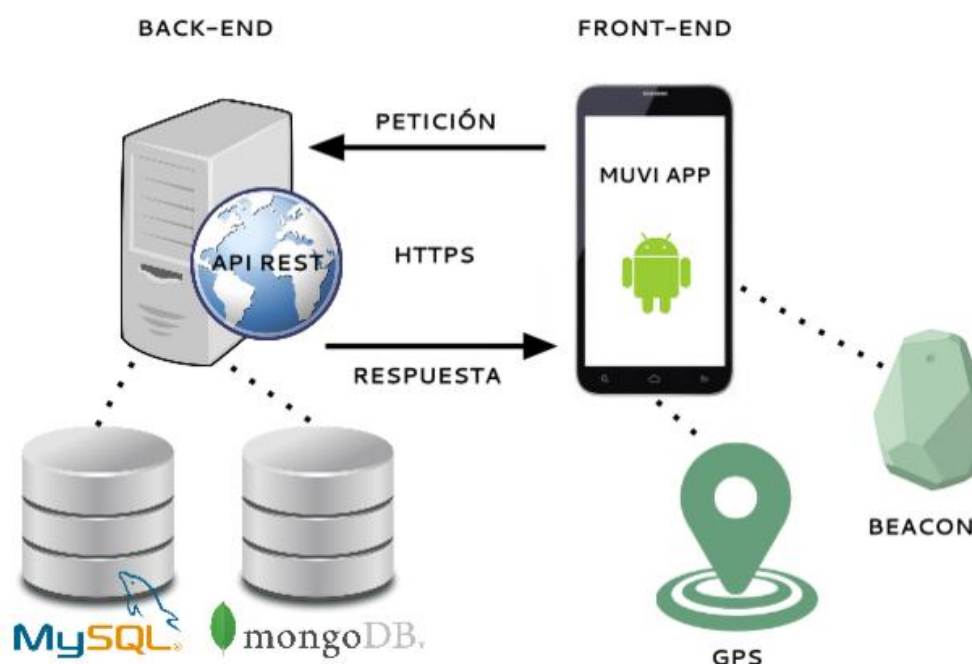


Figura 44: Arquitectura de la aplicación

La arquitectura de nuestra aplicación es una arquitectura *cliente-servidor*. Se trata de un sistema en el que existen *clientes* (*front-end*), que solicitan servicios, y *servidores* (*back-end*), que los proporcionan. Esta comunicación se realiza a través de Internet y, en nuestro caso, el formato de los datos que intercambian es JSON.

En la parte *servidor* o *back-end*, existe una API REST de servicios, los cuales implementan funcionalidades que requieren hacer uso de los recursos del sistema. Se entiende por recurso la información almacenada en la base de datos. Estos servicios se encuentran alojados en un servidor proporcionado por la Facultad de Informática. Los datos son almacenados en dos tipos de bases de datos diferentes: MySQL y MongoDB.

La parte *cliente* o *front-end* corresponde con nuestra aplicación MUVI, que puede ser instalada en cualquier dispositivo Android. Nuestra aplicación hace uso de los dispositivos Beacon,

orientados a la localización en *indoor*. También utiliza el sistema de localización GPS, orientado a la localización en exteriores.

4.1.1 Relación entre *front-end* y *back-end*

En este apartado se explica el uso que la aplicación MUVI (*front-end*) hace de los servicios de la API (*back-end*). El objetivo es plasmar con mayor claridad la relación que existe entre *front-end* y *back-end*, es decir, el funcionamiento de la aplicación un poco más a bajo nivel, antes de explicar cómo se han implementado ambas partes por separado más adelante.

Los servicios que se consumen desde la interfaz de RA corresponden con los siguientes recursos del sistema:

- El recurso de películas se utiliza para consultar la información de una película escaneada cuando el usuario desea ver su información.
- El recurso de valoraciones se utiliza para añadir la valoración de un usuario cuando realiza una valoración de la película escaneada y para consultar las valoraciones que amigos cercanos han realizado de la dicha película.
- El recurso de deseos se utiliza para añadir la película a la lista de deseos del usuario cuando éste la añade a favoritos.
- El recurso de sistema de recomendación se utiliza para estimar lo que a alguno de los amigos cercanos les gustaría la película escaneada cuando éste no ha valorado dicha película con anterioridad.
- El recurso de dispositivos Beacon corresponde con los datos de las conexiones entre usuarios y *beacons* almacenados en la base de datos provenientes del escaneo de *beacons* realizado en el *front-end*, en concreto, en la interfaz de RA. Se utiliza tanto para almacenar cierta información de un usuario y de un *beacon* que están conectados, como para eliminarla si pasa cierto tiempo desde que el dispositivo del usuario escaneó ese *beacon* por última vez.

Los servicios que se consumen desde el resto de interfaces de la aplicación corresponden con los siguientes recursos del sistema:

- El recurso de usuarios se utiliza para guardar los datos de un usuario si crea una cuenta a través de la interfaz de Registro o para consultar los datos del mismo si desea acceder a la aplicación a través de la interfaz de Inicio de Sesión.
- El recurso de películas se utiliza consultar la información de una película cuando se necesita mostrar dicha información en la interfaz de Ficha de Película.
- El recurso de deseos se utiliza para consultar las películas que le gustan al usuario cuando se necesita mostrar dichas películas en la interfaz de Me Gusta. También se

utiliza en la interfaz de Gente Cercana para consultar que películas le gustan a cada uno de los usuarios cercanos.

- El recurso de GPS corresponde con los datos de ubicación almacenados en la base de datos provenientes del uso del sistema de localización GPS en el *front-end*. Se utiliza en la interfaz de Gente Cercana, tanto para añadir la ubicación de usuarios cercanos, como para consultar los usuarios cercanos cuando se necesita mostrar dichos usuarios.

4.2 Implementación del *back-end*

Los módulos de los que se compone el *back-end* son los siguientes:

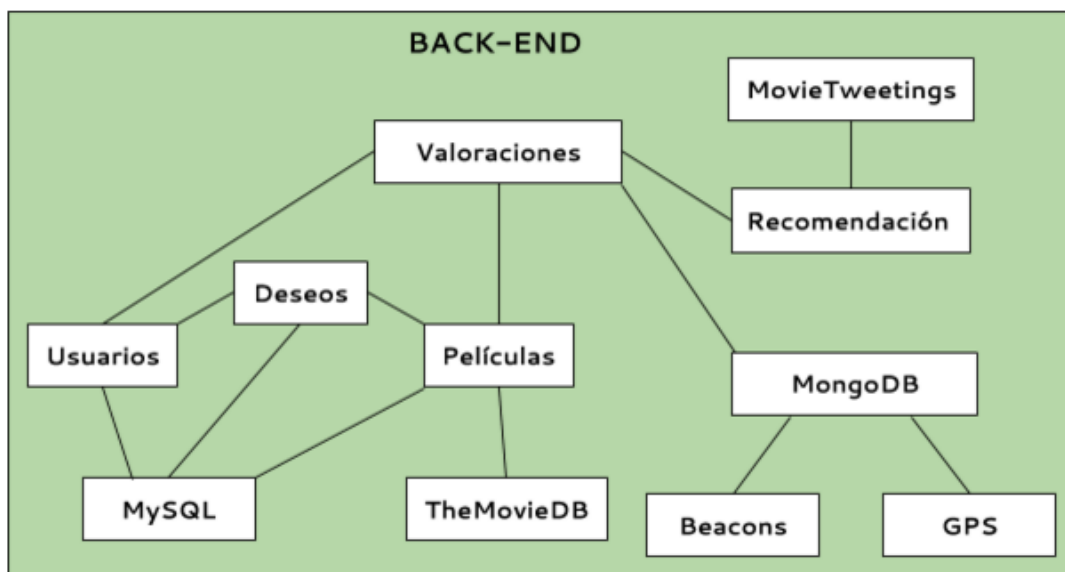


Figura 45: Módulos del back-end

Las relaciones entre los módulos se representan con una línea que los une. A continuación, se explican cada una de estas relaciones:

El módulo de usuarios gestiona la información de los usuarios almacenada en la base de datos relacional en MySQL. El módulo de películas obtiene la información de las películas de una fuente de información externa, en nuestro caso TheMovieDB, y por eso está relacionado con el módulo de TheMovieDB. La información que obtiene de TheMovieDB, la gestiona, almacenando parte de ella en la base de datos relacional en MySQL. El módulo de deseos gestiona las listas de películas deseadas de los usuarios. Es por esta razón que está relacionado con ambos. Los módulos que almacenan la información en la base de datos relacional MySQL es por eso que están relacionados con el módulo de MySQL.

El módulo de valoraciones gestiona la información de las valoraciones que realizan los usuarios de nuestro sistema sobre las películas. Es por esta razón que está relacionado, tanto con los

usuarios, como con las películas. Por otro lado, estas valoraciones realizadas por los usuarios de nuestro sistema, junto con las valoraciones del *dataset* MovieTweatings, se utilizan en nuestro sistema de recomendación. Por ello, tanto el módulo de valoraciones, como el módulo de MovieTweatings, están relacionados con el módulo de recomendación.

En un principio se diseñó el *back-end* pensando en que el sistema de recomendación se basara en datos de valoraciones que se almacenaran en MongoDB. Posteriormente, la tecnología con la que se implementó el sistema de recomendación no permitía acceder a los datos directamente desde MongoDB, sino que requería de un fichero de datos. De este hecho se deriva que las valoraciones de nuestro sistema se almacenen y consulten en MongoDB y que ambos módulos estén relacionados. Se explica con más detalle en el apartado *4.2.3 Sistema de recomendación* de este documento.

La funcionalidad que interactúa con los sistemas de localización, tanto dispositivos Beacon, como GPS, se implementa en el *front-end*. Sin embargo, los datos que es necesario persistir se envían al *back-end* y se almacenan en MongoDB. Es por esta razón que existen los módulos de *beacons* y *GPS*. El módulo de beacons gestiona los datos de las conexiones entre dispositivos Beacon y usuarios, almacenados en MongoDB. El módulo de GPS gestiona los datos de ubicación de los usuarios obtenidos a través del sistema de localización GPS, almacenados en MongoDB, y obtiene, a partir de ellos, los usuarios cercanos dentro de un rango de distancia. Los módulos que almacenan la información en la base de datos orientada a documentos MongoDB es por eso que están relacionados con el módulo de MongoDB.

A continuación, se detalla cómo se ha implementado este *back-end*, incluyendo la explicación de la arquitectura de los servicios y la documentación de los mismos, así como las diferentes tecnologías utilizadas en su desarrollo, para la persistencia de los datos y en el sistema de recomendación. También se explican las tareas y problemáticas derivadas del servidor experimental proporcionado por la Facultad de Informática.

4.2.1 Arquitectura de servicios

REST (REpresentational State Transfer) es un tipo de arquitectura de servicios que se apoya en el estándar HTTP. Se definió en el año 2000 por Roy Fielding, coautor de la especificación HTTP.

Existen varios niveles de cumplimiento de los principios REST. Se describen en un modelo llamado *Richardson Maturity Model* (en adelante RMM) (31), cuyo autor es Leonard Richardson, padre de la arquitectura orientada a recursos.

- Nivel 0 – Swamp of POX: Los servidores utilizan HTTP, pero únicamente como medio de transporte. Ejemplo: SOAP

- Nivel 1 – Recursos: Los servicios utilizan las URIs HTTP para distinguir los recursos del sistema. Se entiende por recurso la información a la que se quiere acceder, modificar o borrar, independientemente de su formato. Estas URIs permiten identificar de forma única el recurso.
- Nivel 2 – Verbos HTTP: Los servicios se aprovechan de las características nativas de HTTP como cabeceras, códigos de estado, métodos, etc. Ejemplo: Amazon S3
- Nivel 3 – Hypermedia: Los servicios utilizan HATEOAS (Hypermedia as the Engine of Application State). HATEOAS es una restricción que implica que el *cliente* interactúa con el *servidor* a través de hipermedias proporcionadas dinámicamente por el propio servidor.

Nuestra API se encuentra en el nivel 2. Siendo estrictos con la definición original de REST realizada por Roy Fielding, nuestra API no sería REST, porque no utilizamos hipermedias (32). Sin embargo, son muchas las grandes empresas (Flickr, Twitter, Git, Amazon S3...) que dicen tener una API REST y tampoco respetan la definición original.

Las características de nuestra API serían las siguientes:

- Uso correcto de URIs (Nivel 1 del modelo RRM):
 - Los nombres de URI no implican acciones, es decir, no se utilizan verbos.
 - Cada URI es única y no existen dos URIs que identifiquen el mismo recurso.
 - Son independientes del formato.
 - Mantienen una jerarquía lógica.
 - Los filtrados de información de un recurso no se hacen en la URI.
- Uso correcto de HTTP (Nivel 2 del modelo RRM):
 - Se utilizan los verbos HTTP apropiados para cada acción concreta (GET, POST, DELETE...).
 - Se utilizan los códigos de error/estado HTTP en las respuestas a las peticiones.
 - Se especifica el formato del recurso a través de HTTP.

La principal referencia a la hora de entender y aplicar estas características en nuestra API ha sido un artículo de Asier Marqués en el que resume algunos de los conceptos más importantes que caracterizan a las APIs REST (33).

Nuestra API REST de servicios implementa servicios relacionados con los distintos recursos del sistema: los usuarios, las películas, las valoraciones de los usuarios, los deseos del usuario, el sistema de recomendación, la geolocalización GPS y los dispositivos Beacon.

Los servicios relacionados con los usuarios son:

- Login de un usuario: comprueba los datos de acceso de un usuario.

- Registro de un usuario: registra un nuevo usuario en nuestro sistema.
- Buscar un usuario: busca un usuario por su nombre.

Los servicios relacionados con las películas son:

- Buscar información de una película: busca la información de una película por su título.
- Listar películas: lista las películas que coinciden con un título (máximo 5 películas).

Los servicios relacionados con las valoraciones de los usuarios son:

- Añadir una valoración de un usuario: añade una valoración de una película realizada por un usuario al sistema.
- Mostrar una valoración de un usuario: muestra la valoración de una película realizada por un usuario.

Los servicios relacionados con los deseos del usuario son:

- Añadir un deseo de un usuario: añade una película a la lista de deseos de un usuario.
- Listar los deseos de un usuario: lista las películas de la lista de deseos de un usuario.
- Eliminar un deseo de un usuario: elimina una película de la lista de deseos de un usuario.

Los servicios relacionados con el sistema de recomendación son:

- Estimar una recomendación: estima la valoración que un usuario daría a una película.

Los servicios relacionados con la geolocalización GPS son:

- Añadir ubicación de un usuario: añade la ubicación de un usuario al sistema.
- Eliminar ubicación de un usuario: elimina la ubicación de un usuario del sistema.
- Listar usuarios cercanos: lista los usuarios cercanos con respecto a la ubicación actual.

Los servicios relacionados con los dispositivos Beacon son:

- Asociar un usuario a un *beacon*: asocia un usuario a un dispositivo Beacon cuando se conecta a dicho dispositivo.
- Elimina un usuario de un *beacon*: elimina un usuario de un dispositivo Beacon una vez ha pasado cierto tiempo desconectado.
- Listar los usuarios de un *beacon*: Lista los usuarios conectados a un mismo dispositivo Beacon.

En el ANEXO I de este documento se encuentra una documentación más detallada y más técnica de nuestra API REST de servicios.

En el desarrollo de nuestra API REST se ha utilizado el lenguaje de programación Java y el *framework* Jersey. Jersey es un *framework open source* para el desarrollo de servicios web RESTful en Java. Se trata de una implementación de JAX-RS (Java API for RESTful Web

Services) con características adicionales y proporciona anotaciones que facilitan la creación de servicios web RESTful.

Para la gestión de la persistencia en Java se ha seguido la especificación JPA utilizando el *framework* Hibernate. La relación que existe entre JPA e Hibernate es que este último implementa como parte de su código especificaciones de JPA. A parte de implementar del estándar JPA, también añade más funcionalidad. Este *framework* permite gestionar la capa de persistencia a través de ficheros.xml o con anotaciones. En nuestro caso hemos utilizado anotaciones. Para su configuración hemos hecho uso del fichero `persistence.xml`, el cual contiene las unidades de persistencia, el tipo de conexión con la base de datos y otros parámetros propios de la implementación JPA.

Como IDE (Entorno de Desarrollo Integrado) para los servicios utilizamos Eclipse con el *plugin* WTP (Web Tools Platform). La razón de utilizar este IDE ha sido que ya teníamos experiencia previa con él. En Eclipse también integramos Maven y Tomcat. Por un lado, Maven es una herramienta para la gestión y construcción de proyectos Java. La principal ventaja de usarla ha sido que nos ha facilitado la gestión de dependencias. Esta herramienta funciona de tal forma que, si se le indica qué librerías externas se necesitan, es capaz de buscarlas ella misma y descargarlas por nosotros. Por otro lado, Tomcat es un contenedor de *servlets*. Lo hemos utilizado para ejecutar nuestros servicios web, tanto en local, como en el servidor proporcionado por la Facultad de Informática.

4.2.2 Base de datos

Nuestra base de datos se trata de una base de datos híbrida en la que se usa tanto el modelo relacional (MySQL), como no relacionales (MongoDB).

Como sistema gestor de base de datos relacional hemos utilizado MySQL. Las razones de elegir este gestor son que es *open source* y uno de los más populares. Como herramienta visual de administración a nivel local hemos utilizado MySQL Workbench.

Por otro lado, MongoDB es un sistema de base de datos NoSQL orientado a documentos. Se trata de una base de datos ágil que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan, incluyendo la funcionalidad de las bases de datos tradicionales. Proporciona escalabilidad, rendimiento y gran disponibilidad.

La necesidad de usar MongoDB se derivó, en un principio, de tener que integrar las valoraciones del *dataset* de MovieTweatings con las valoraciones de los usuarios de nuestro propio sistema. El esquema relacional no se ajusta al esquema que necesitamos para realizar dicha integración. MongoDB, sin embargo, es flexible en ese sentido.

Además, MongoDB es una buena opción para gestionar datos geográficos sencillos, ya que guardando coordenadas geográficas (latitud, longitud) permite realizar consultas basadas en el posicionamiento, ya sea por proximidad, donde obtendremos los documentos ordenados por proximidad (de más cercano a más distante) a un punto geográfico, o consultas acotadas, donde obtendremos los documentos que están dentro de un área (rectángulo, círculo o polígono). Esta utilidad de MongoDB nos permite saber que amigos se encuentran próximos a nosotros a la hora de usar la aplicación.

Por otro lado, MongoDB proporciona un mejor rendimiento a la hora de hacer una gran cantidad de consultas en poco tiempo, necesario durante el escaneo de dispositivos Beacon.

Modelo relacional en MySQL

El diseño de nuestra base de datos relacional en MySQL corresponde con las siguientes tablas:

- Tabla *Usuario*: contiene la información de los usuarios que se registran en nuestra aplicación. Los atributos *Email* y *Alias* serán únicos.
- Tabla *Película*: contiene la información de las películas que el usuario busca a través de nuestra aplicación. Solo se almacena la ID de IMDb y el nombre de la película.
- Relación *Desea*: muestra la relación entre un usuario y las películas que ha guardado en su lista de deseos.

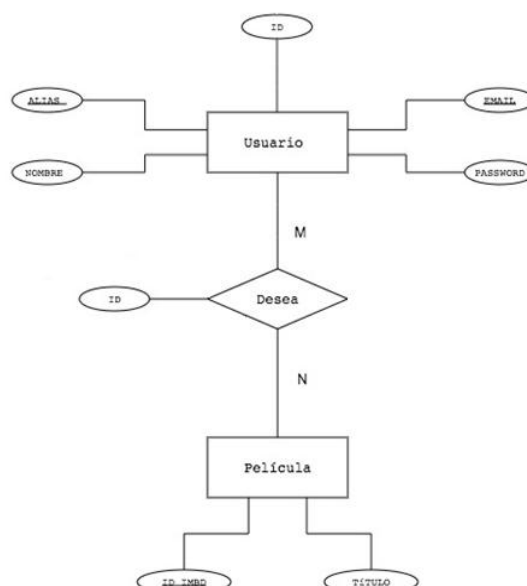


Figura 46: Diseño E/R de la base de datos MySQL

Colecciones en MongoDB

Estructura colección *Beacons*. almacena los usuarios que están dentro del alcance de un dispositivo Beacon.

```
{
  "_id": <ObjectID>,
  "id_usuario": <NumberLong>,
  "id_beacon": <String>,
  "name": <String>
}
```

Estructura colección *GPS*. almacena las coordenadas geoestacionarias de los usuarios conectados a la aplicación.

```
{
  "_id": <ObjectID>,
  "id_usuario": <NumberLong>,
  "name": <String>,
  "loc": {
    "lat": <Latitude>,
    "lon": <Longitude>
  }
}
```

Estructura colección *Valoraciones*. almacena las valoraciones que dan los usuarios a las diferentes películas, así como las valoraciones del *dataset* MovieTweetintg. El valor de la clave *origen* hace referencia a la procedencia de estas valoraciones, es decir, nuestro sistema o dicho *dataset*.

```
{
  "_id": <ObjectID>,
  "origen": <String>,
  "id_usuario": <NumberLong>,
  "id_pelicula": <NumberLong>,
  "valoración": <NumberLong>
}
```

4.2.3 Sistema de recomendación

En un principio íbamos a utilizar el proyecto jCOLIBRI para el desarrollo de nuestro sistema de recomendación, pero al intentar incorporarlo en nuestra aplicación descubrimos que eran incompatibles debido a las diferentes versiones de Java. JCOLIBRI necesita la versión 6 para poder compilar y funcionar, en cambio nuestro proyecto funciona bajo la versión 7 de Java. Por este motivo, tuvimos que buscar alternativas.

Finalmente, en la implementación de nuestro sistema de recomendación se ha realizado con la API de Apache Mahout. Esta API incluye algoritmos de aprendizaje automático de distintas áreas, incluyendo algoritmos de recomendación. La técnica que hemos utilizado es el filtrado colaborativo basado en el usuario.

Para el desarrollo de nuestro servicio de recomendación partimos de un artículo de Apache Mahout (34). La única diferencia con respecto a lo que se indica en dicho artículo es que nosotros utilizamos el método `estimatePreference(long userID, long itemID)` en lugar de `recommend(long userID, int howMany)` porque nuestra necesidad es la de estimar lo que le gustaría a un usuario concreto una determinada película. La estimación se produce si dicho usuario aún no ha valorado esa película y si hay datos suficientes para realizar dicha estimación. Mediante el método `recommend(long userID, int howMany)` sería posible conseguir una lista de las N películas que más le gustarían a un usuario concreto ordenadas de mayor a menor recomendación. Esta funcionalidad podría formar de una posible ampliación de la aplicación en un futuro.

En el caso de que no haya datos suficientes para realizar la estimación, se produce lo que se conoce como *cold start* o *arranque en frío*. Se trata de un problema común en los sistemas de recomendación. Nuestra solución para este problema consiste en calcular la media de las valoraciones existentes de la película sobre la que se desea hacer la recomendación. Para evitar en la medida de lo posible esta situación, se integran los datos del *dataset* de MovieTweatings (35) en nuestro sistema. Este *dataset* contiene valoraciones de películas que se obtienen de los *tweets* de Twitter.

El uso de la API de Apache Mahout requiere almacenar las valoraciones en un fichero, en nuestro caso un `.csv`. En este fichero se almacenan tanto las valoraciones del *dataset* MovieTweatings, como las de los usuarios de nuestro sistema. El formato en el que se representan, correspondiendo con el formato de los ficheros `.csv`, es en forma de tabla, de tal forma que las columnas (ID del usuario, ID de la película, valoración) se separan con comas y las filas por saltos de línea.

La colección *Valoraciones* de MongoDB contiene la misma información que este fichero `.csv`. En un principio, se utilizaba únicamente la colección de MongoDB para almacenar la información de las valoraciones. Posteriormente, al conocer el requerimiento de un fichero (en nuestro caso `.csv`) para poder utilizar la API de Apache Mahout para la recomendación, se duplicó esta información en este fichero `.csv` y se amplió la funcionalidad de forma que cuando un usuario realizara una nueva valoración, se añadiera tanto a la colección de MongoDB, como al fichero `.csv`. MongoDB se mantuvo porque es necesario hacer consultas de las valoraciones y hacerlas sobre un fichero `.csv` que contiene gran cantidad de datos sería muy ineficiente.

Para incorporar los datos de las valoraciones del *dataset* de MovieTweatings, tanto a la colección de *Valoraciones* de MongoDB, como al fichero `.csv`, se realizaron dos scripts en el lenguaje de programación Python: uno que lee los datos del archivo `.dat` original del *dataset*

MovieTweatings y los guarda en nuestra colección `valoraciones` de MongoDB, y otro que crea un archivo `.csv` a partir de los datos del archivo `.dat` original de MovieTweatings.

4.2.4 Servidor

Durante la realización de este TFG hemos formado parte de un programa piloto de la Facultad de Informática, teniendo disponible un servidor experimental montado en una máquina virtual, con 500 Mb de disco duro, 500 Mb de memoria RAM y Ubuntu 14.0 como sistema operativo.

Para el correcto funcionamiento de nuestra aplicación tuvimos que instalar y configurar en el servidor tanto SQL Server como MongoDB.

A la hora del despliegue de nuestros archivos WAR nos hemos encontrado problemas con incompatibilidades como, por ejemplo, que en el servidor está instalado Java 8 y Mahout no es compatible a día de hoy con esta versión de Java, por lo tanto, hemos tenido que cambiar de versión en el servidor para tener compatibilidad.

Otro problema que hemos tenido que solventar es una limitación de 50 Mb de tamaño en los archivos WAR subidos al Tomcat. Al principio no tuvimos problemas con este límite de tamaño, pero llegado un momento en el que la aplicación tenía ya un desarrollo bastante avanzado, con varias librerías incluidas, nuestros archivos ocupaban más de lo permitido, por lo que hemos tenido que modificar los archivos de configuración de Tomcat para aumentar el tamaño máximo permitido.

Respecto a Mahout hemos tenido problemas con dependencias que tiene esta librería, interfiriendo con Hibernate y con Jersey. Esta situación ha sido solucionada con la ayuda de las exclusiones que permite realizar Maven y aligerando la librería de Mahout con sólo lo imprescindible para el funcionamiento de nuestro servicio de recomendación.

Para este servicio de recomendación, en un primer momento hemos utilizado un *dataset* con más de cuarenta mil valoraciones de películas. En nuestras pruebas en local no hemos tenido ningún problema, pero a la hora de probarlo en el servidor hemos tenido que reducir el número de valoraciones a unas tres mil debido a la poca memoria RAM disponible en la máquina virtual, que producía que se quedara bloqueada a la hora de ejecutar el servicio de recomendación, provocando la parada de Tomcat.

4.2.5 Fuente de información sobre películas

La información de las películas que se le muestra al usuario (título, sinopsis, tráiler, etc) se obtiene de TheMovieDB a través de su API. TheMovieDB es una de las mayores bases de datos de películas disponibles en Internet. Proporciona un extenso manual y diversos ejemplos que

nos han servido de gran ayuda a la hora de implementar nuestros servicios. Para obtener la información de las películas hemos utilizado tres de los métodos disponibles en su API:

Método para buscar por palabra clave:

`http://api.themoviedb.org/3/search/movie?query=<palabra clave>`

Este método devuelve una lista de todas las películas cuyo título contenga la palabra clave. Nos ha proporcionado la sinopsis, el título y el ID asignado por la API.

Método para buscar por ID:

`http://api.themoviedb.org/3/movie?<ID>`

Este método devuelve información ampliada de la película que coincide con ese ID. Nosotros hemos aprovechado la información sobre los géneros de la película y el ID de la IMDb, el cual hemos utilizado como ID en nuestra base de datos. La razón de utilizar el ID de IMDb como ID de nuestro sistema es porque es el ID que se utiliza en el *dataset* de MovieTweatings.

Método para buscar el tráiler:

`http://api.themoviedb.org/3/movie?<ID>/videos/`

Este método proporciona el tráiler de la película, si existe.

La única información que se almacena en nuestra base de datos es el ID de IMDb y el nombre de la película, el resto de información la obtenemos utilizando los métodos anteriormente descritos cuando el usuario lo requiere.

Antes de decidirnos a utilizar TheMovieDB, decidimos usar una API no oficial de Metacritic de pago, pero con una versión gratuita limitada a 1000 búsquedas, que, a partir del título de una película, devolvía diversa información sobre ella. Sin embargo, en enero recibimos un correo de su creador advirtiéndolo que la API cerraría debido a problemas legales con CBS, dueños de Metacritic, por lo que tuvimos que buscar otras alternativas, y de ahí surgió el uso de TheMovieDB como nuestra fuente de información de películas.

4.3 Implementación del *front-end*

Los módulos de los que se compone el *front-end* son los siguientes:

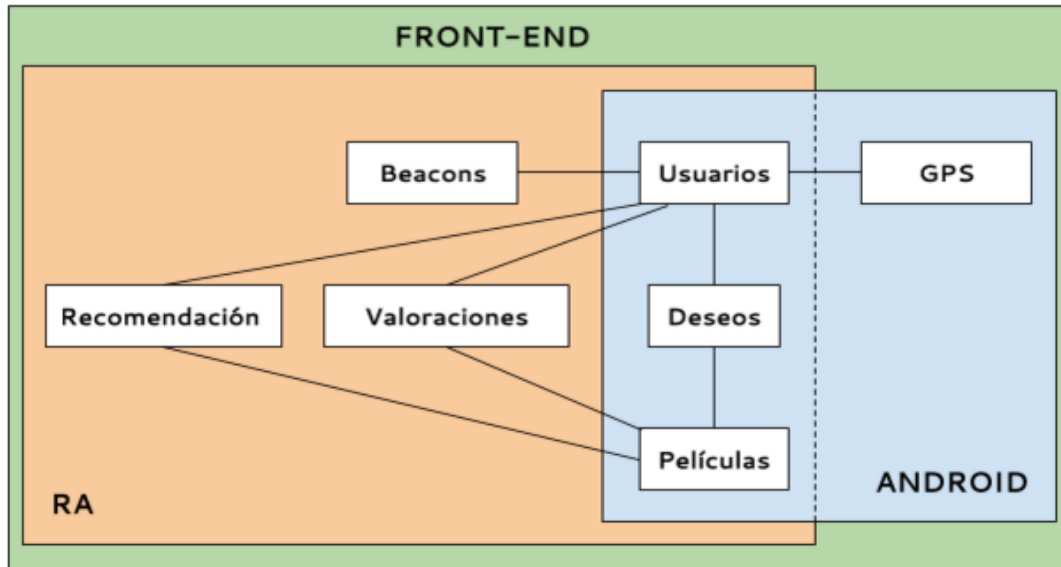


Figura 47: Módulos del front-end

Como en el diagrama de módulos de la Implementación *back-end*, las relaciones entre módulos se representan con una línea que los une.

La RA se relaciona con la Recomendación debido a que la utilizad a la hora de recomendar a un grupo de amigos una película. Con Beacon y Usuarios debido a que cuando se buscan los *beacons* se guarda la información de los usuarios y los *beacons* que están conectados. Se relaciona con Valoración, Deseos y Películas gracias a que cuando se escanea el cartel de una película, muestra la información de la película y tráiler, y permite tanto darle una valoración, como añadirla a la lista de deseos del usuario.

Las interfaces Android se relacionan con Usuarios debido que es necesario hacer el registro y el *login* para entrar en la aplicación. Con Deseos y Películas porque la interfaz de Me Gusta muestra las películas que un usuario ha añadido a la lista de deseos y la interfaz de Ficha de Película muestra toda la información de una película. Por último, se relaciona con GPS debido que se utiliza para mostrar la gente que está cerca de ti en la Interfaz de Gente Cercana.

A continuación, se detalla cómo se ha implementado el *front-end*, incluyendo la explicación de los lenguajes utilizados, tecnologías, así como las diferentes herramientas utilizadas en su desarrollo.

4.3.1 Entorno de desarrollo

Para todo el desarrollo de la parte cliente del proyecto se utilizó el lenguaje Java. Java es el lenguaje que hay que utilizar para programar en Android Studio, por lo que no hubo

mucha elección posible. Además, es un lenguaje que conocemos todos los integrantes del grupo y, por esta razón, aprender a programar aplicaciones móviles para Android ha sido más fácil.

Sin embargo, para poder trabajar con Wikitude tuvimos que utilizar Javascript. Es un lenguaje fácil de aprender, que conocemos casi todos los integrantes y es muy sencillo de utilizar y trabajar con él.

Toda la parte cliente se desarrolló en un mismo entorno de desarrollo. Aunque al principio estuvimos dudando entre Eclipse o Android Studio, finalmente nos decantamos por este último. Android Studio ofrecía más herramientas y una forma más sencilla de trabajar.

Android Studio es un entorno de desarrollo integrado para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. Elegimos utilizar Android Studio por su integración con las APIs de Android y el fácil manejo que tiene para desplegar y testear las aplicaciones en los dispositivos móviles. Además de esto, Android Studio ofrece varias ventajas:

- Renderización en tiempo real.
- Consola de desarrollador: consejos de optimización, ayuda para la traducción, estadísticas de uso.
- Soporte para construcción basada en Gradle.
- Refactorización específica de Android y arreglos rápidos.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones, y otros problemas.
- Plantillas para crear diseños comunes de Android y otros componentes.

4.3.2 Realidad Aumentada

A la hora de desarrollar la 3.4.4 Interfaz de Realidad Aumentada teníamos varias herramientas posibles. Estuvimos bastante tiempo investigando cuál sería mejor para el proyecto y haciendo pruebas con Vuforia y Wikitude. Finalmente, decidimos usar la tecnología Wikitude en vez de Vuforia debido a su fácil integración con Android Studio, porque dispone de un lenguaje de programación sencillo (JavaScript) y porque Vuforia, al diseñarse en Unity, está más orientado a juegos que a aplicaciones, como la de este proyecto, y permite reconocer imágenes tanto en local como en la nube.

Decidimos usar el reconocimiento en la nube de las imágenes debido a que la Target API de Wikitude es muy sencilla de utilizar, ya que se pueden añadir muchos carteles muy fácilmente, y la base de datos de la aplicación se actualiza de forma automática. Sin embargo, nos

encontramos con el problema de que, al estar el reconocimiento en la nube limitado a 1000 reconocimientos al mes (en la versión gratuita) y tener que hacer tantas pruebas para ver si funcionaba correctamente la aplicación, rápidamente se nos prohibía utilizar más reconocimientos porque habíamos superado el límite.

Por ese motivo, decidimos realizar el reconocimiento de imágenes en local, ya que no tiene límite. Sin embargo, no se actualiza automáticamente la aplicación y es necesario descargar un archivo `.wtc` de la API de Wikitude con las imágenes que queremos reconocer y añadirlo a la aplicación cada vez que queremos cambiar los carteles de las películas que reconocemos.

Otro inconveniente que encontramos fue la reproducción de video en la RA. Uno de los objetivos de la interfaz RA era mostrar el tráiler de la película que anunciaba el cartel. Sin embargo, leyendo la documentación de Wikitude, comprobamos que sólo se pueden crear 4 objetos de video debido al coste de carga de estos videos. Es por ello que nuestra interfaz de RA se vio limitada a reconocer únicamente cuatro carteles de película, siendo imposible así realizar el objetivo de guardar toda una cartelera.

Otro problema con la reproducción de los tráileres está relacionado con la forma en la que se guardan los videos y los objetos de reconocimiento. La primera idea era un único objeto `AR.Trackable2DObject` que recibía el nombre del póster y llamaba a los servicios web para mostrar toda la información por pantalla. Sin embargo, al tener que crear un objeto de video (`AR.VideoDrawable`) para cada tráiler, fue necesario crear un objeto de reconocimiento para cada cartel, lo que provoca mucho código repetido y no ser tan fácilmente ampliable. Una posible solución a estos problemas es no mostrar el video en la RA y que, al pulsar el botón de video, se dirija la reproducción del tráiler a YouTube.

4.3.3 Tecnologías

Por otro lado, parece importante explicar qué otras tecnologías se usaron para la creación de las interfaces de usuario.

Al ser una aplicación con interfaces parecidas, que, además, el usuario puede ir de una a otra rápidamente, tuvimos que encontrar la manera de que las llamadas al servidor se hicieran de forma eficiente y sin que el usuario lo notara. Para ello, utilizamos la tecnología Fragment. Esta tecnología nació cuando empezaron a aparecer dispositivos de gran tamaño tipo tablet, el equipo de Android tuvo que solucionar el problema de la adaptación de la interfaz gráfica de las aplicaciones a ese nuevo tipo de pantallas. Una interfaz de usuario diseñada para un teléfono móvil no se adaptaba fácilmente a una pantalla varias pulgadas mayor. La solución a esto vino en forma de un nuevo tipo de componente llamado Fragment.

Un *fragment* no puede considerarse ni un *control* ni un *contenedor*, aunque se parecería más a lo segundo. Un *fragment* podría definirse como una porción de la interfaz de usuario que puede añadirse o eliminarse de la interfaz de forma independiente al resto de elementos de la actividad, y que por supuesto puede reutilizarse en otras actividades. Esto, aunque en principio puede parecer algo trivial, nos va a permitir poder dividir nuestra interfaz en varias porciones de forma que podamos diseñar diversas configuraciones de pantalla, dependiendo de su tamaño y orientación, sin tener que duplicar código en ningún momento, sino tan sólo utilizando o no los distintos fragmentos para cada una de las posibles configuraciones.

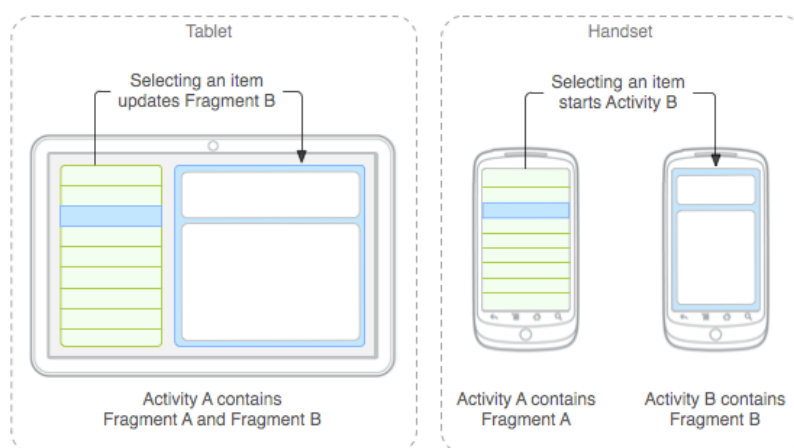


Figura 48: Ejemplo de uso de los fragmentos de Android

El uso de *fragmentos* nos ha ayudado bastante a la hora de gestionar las IU, teniendo una actividad principal y después varios *fragmentos* que se combinan para crear todo el conjunto de interfaces. Los hemos utilizado en las interfaces de Inicio, Me gusta y Ficha de Película. En nuestro caso lo que tenemos es un contenedor que tiene el menú desplegable y después los diversos fragments que aparecen o desaparecen en función de lo que el usuario haya pulsado en el menú.

4.3.3.1 Localización indoor con dispositivos Beacon

La localización *indoor* con dispositivos Beacon se ha implementado con el SDK de Konkakt.io, que es una compañía que ofrece servicios de seguridad y configuración de dispositivos Beacon, tanto hardware, como software (36).

El uso de *beacons* en nuestra aplicación se deriva de la necesidad de localizar dispositivos móviles cercanos en *indoor*. Los *beacons* permiten al usuario ubicarse en sitios donde el GPS no es tan útil, por ejemplo, en un cine.

La funcionalidad asociada a estos dispositivos Beacon está asociada a la interfaz de RA. Esta funcionalidad consiste en que cuando el usuario entra en esta interfaz se inicia el escaneo de *beacons* de forma instantánea y permanece constante hasta que el usuario sale de dicha interfaz. Si durante el escaneo se encuentra uno de estos dispositivos, se asocia dicho usuario a dicho *beacon* en el sistema. De esta forma, cuando se muestran los amigos cercanos a un usuario en la RA, se consultan los usuarios que están asociados al mismo dispositivo Beacon que el usuario en cuestión.

En el desarrollo de esta funcionalidad, partimos de una demo realizada por José Luis Jorro Aragonese. En ella se utiliza el SDK para Android de Konkakt.io. Este SDK proporciona componentes para construir aplicaciones utilizando *beacons* y cubre las siguientes áreas de funcionalidad:

- El *ranging/monitoring* del dispositivo.
- La conexión del dispositivo.
- La comunicación con la API REST de Konkakt.io.
- Las acciones del dispositivo.

El *manager* utilizado para el escaneo de *beacons* ha sido ProximityManager. Este *manager* presenta ciertos inconvenientes con respecto a la otra alternativa (KonkaktProximityManager), pero decidimos no actualizarlo debido a que llevábamos cierto retraso en el desarrollo de la funcionalidad asociada a estos dispositivos Beacon. Sería una posible mejora de futuro.

Por otro lado, el método utilizado para este escaneo ha sido *ranging*. Consiste en escanear dispositivos Beacon de forma instantánea y constante una vez iniciado el escaneo. En nuestra aplicación este escaneo se inicia al entrar en la interfaz de RA.

La demo inicial de J. L. Jorro permitía escanear *beacons* en sus dos formatos: iBeacon y Eddystone. Esta demo no era del todo funcional, ya que no realizaba el escaneo correctamente. En el proceso de solución de este problema, limitamos esta característica, y nuestra aplicación actualmente sólo escanea dispositivos Beacon en formato Eddystone. Esta sería otra posible mejora de futuro.

4.3.3.2 Geolocalización con GPS

Es una tecnología de geoposicionamiento utilizando la red de satélites de GPS, la cual, mediante cálculos trigonométricos da una posición con componentes de latitud y longitud, con una precisión de unos pocos metros.

Este apartado en un principio no fue primordial, sino que fue una salvaguarda por si no se podía lograr un correcto funcionamiento en el uso del servicio Beacon de Kontakt.io, pero al final, introdujimos esta funcionalidad para cuando estuviéramos fuera del alcance de algún Beacon. Así distinguimos outdoor (cuando estemos fuera del alcance de un Beacon) e indoor (cuando estemos dentro del alcance de un Beacon).

Implementamos un servicio en segundo plano que nos proporciona nuestras coordenadas geoespaciales mediante el uso de datos móviles, WiFi o sistema GPS. Dicho servicio dependerá de la precisión de nuestra posición, siendo mayor o menor dependiendo de nuestro tipo de conexión.

Este servicio es utilizado en la interfaz gente cercana para proporcionarnos una lista de usuarios cercanos a nosotros, mostrándonos las películas que les gustan a cada uno de ellos.

4.3.4 Herramientas

A la hora de desplegar la aplicación para poder probarla, algunos integrantes del grupo tuvimos problemas, ya que no disponíamos de móvil Android. Por esta razón, tuvimos que investigar y utilizar una herramienta que nos permitiese desplegar la aplicación en el ordenador. Aunque Android Studio tiene esta posibilidad, el rendimiento es bastante malo y decidimos utilizar Genymotion. Se trata un emulador móvil mucho más eficiente que el que por defecto ofrece Android Studio. Utiliza VirtualBox para crear los diferentes dispositivos móviles (emulados). Además, Android Studio reconoce el móvil de GenyMotion cuando éste está corriendo. Nos ha sido muy útil a la hora de *debugear* y testear, ya que utilizando esta herramienta no ha sido necesario reiniciar el ADB cada vez que ejecutábamos la aplicación desde Android Studio.

4.4 Pruebas de arquitectura

En este apartado se explican las diferentes pruebas llevadas a cabo como toma contacto con las tecnologías y/o funcionalidades del proyecto que eran totalmente nuevas para nosotros. Estas pruebas se agrupan en cinco grupos: de RA, de servicios web, con Hibernate/JPA, de los servicios de redes sociales y de geolocalización.

4.4.1 Pruebas de realidad aumentada

El objetivo de realizar pruebas con varias tecnologías de RA es conocer cómo funcionan y qué nos ofrecen, para saber con cuál nos puede resultar más fácil implementar la RA de nuestra aplicación o cuál se ajusta más a nuestras necesidades.

Pruebas en Unity con Vuforia

- Pruebas de reconocimiento de texto

Una de las ideas al principio del desarrollo del proyecto era que, en vez de reconocer la imagen de un cartel, la aplicación pudiera reconocer el nombre de la película a través del texto. Para ello, realizamos una prueba con el reconocedor de texto de Vuforia.

Esta funcionalidad de Vuforia se encuentra, una vez instalado el paquete de Vuforia en Unity, en una escena en la carpeta *assets > Scenes > Vuforia-TextReco*. Para poder ejecutarla en el ordenador, es necesario que éste tenga cámara o que se ejecute en un teléfono utilizando la aplicación de Unity para dispositivos móviles.

En las pruebas se muestra una pequeña franja en la que añadir texto, y las palabras en inglés con fuente normal se reconocen fácilmente, especialmente en fondo claro. Sin embargo, al probar con carteles como los de *Star Wars* y *Planet 51*, que tienen unas fuentes muy diferentes, la aplicación es incapaz de reconocer los nombres.

Tras estas pruebas, descartamos la idea del reconocimiento de texto para los carteles, ya que las fuentes son un problema, debido a que muchas son irreconocibles para los reconocedores de texto.

- Prueba de reconocimiento en local de una imagen

La prueba de reconocimiento en local la realizamos siguiendo un tutorial del blog *Emiliusvgs* (37). El tutorial está suficientemente completo y no tuvimos mayores problemas para implementar esta prueba.

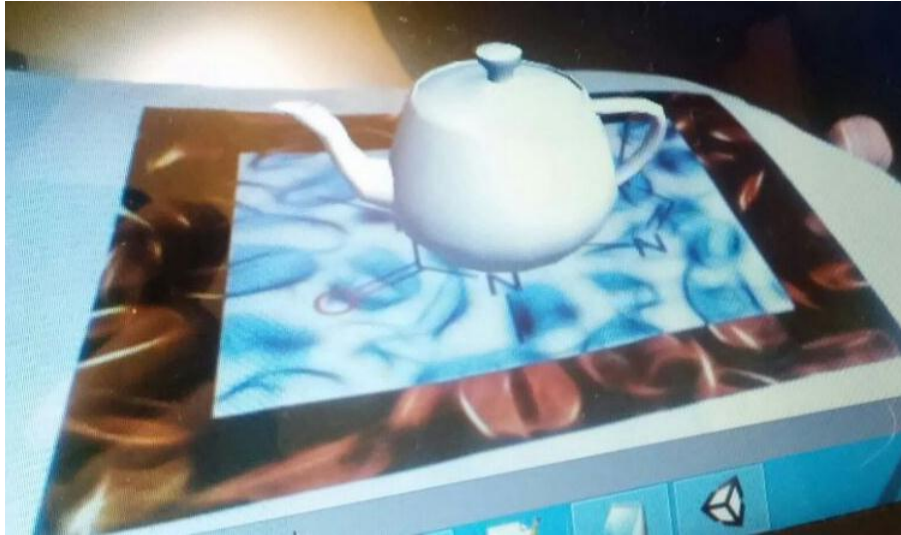


Figura 49: Prueba de reconocimiento local con Vuforia

- Prueba de reconocimiento en la nube de imágenes

Comenzamos siguiendo un artículo de la página de Vuforia en el que se explica cómo crear una app sencilla de reconocimiento en la nube con Unity (38). Siguiendo este tutorial nos encontramos con las siguientes dificultades:

- La utilización de Vuforia en Unity tiene una restricción. Si nuestro sistema operativo es Windows, el editor para Unity que instalemos debe ser necesariamente de 32 bits. Sin embargo, si se trata de un MAC OS, debe ser necesariamente de 64 bits.
- Además de importar el paquete de Vuforia para Unity (`vuforia-unity-5-0-6`), como se indica, también hay que importar el de Cloud Recognition (`CloudReco-5-0-5`) debido a que en el código se implementa la interfaz de `ICloudRecoEventHandler`. Esto último no se especifica en el tutorial.
- En la versión del SDK 4.0, la clase `ImageTracker` ha sido reemplazada por `ObjectTracker`. En el código de ejemplo del tutorial es necesario reemplazar una por otra si se usa la versión 5 del SDK de Vuforia.
- Para ejecutar una aplicación en Vuforia es necesario solicitar una clave de licencia, la cual se puede conseguir de forma gratuita en el portal para desarrolladores de la página web de Vuforia (39).

Vuforia permite la creación de una base de datos en la nube donde se pueden guardar las imágenes que se quieren reconocer, y pone a disposición del usuario claves de acceso para añadirla a la app en Unity. La licencia gratuita permite guardar 1000 imágenes en la base de datos y el mismo número de reconocimientos al mes.

Al añadir una imagen a la base de datos es posible añadir un archivo de metadatos asociado a la imagen. Para la prueba creamos un archivo de metadatos para cada película únicamente con el nombre de ésta. Al tener algunos miembros experiencia a la hora de trabajar con Unity, la prueba fue más sencilla de realizar que si nadie hubiera trabajado antes en Unity. El proceso que se realizó fue el siguiente:

- Utilizando la escena básica de Vuforia *Cloud Reco*, creamos un objeto que aparecerá encima de la imagen reconocida.
- A este objeto le añadimos un script llamado *NameManager*, que recoge el metadata de la imagen y actualiza el objeto anterior con el nombre obtenido en el archivo.



Figura 50: Prueba de reconocimiento en la nube con Vuforia

Pruebas en Android con Wikitude

- Prueba de reconocimiento en local de una imagen

Nuestra primera toma de contacto con el SDK de Wikitude fue mediante varios artículos del blog Desarrollo Libre (40). Estos ejemplos son en local, están implementados con la versión 3 del SDK de Wikitude para Android y utilizan la API de JavaScript. Para ello utilizamos el entorno de desarrollo Eclipse con el *plugin* ADT. Las principales dificultades que nos encontramos tuvieron que ver con nuestra falta de conocimientos en el desarrollo de Android. Algunas de esas dificultades fueron:

- Al importar el código de ejemplo nos aparecía el error "Unable to resolve target 'android-XX' ". Esto ocurría porque la versión de la API del proyecto en Eclipse no se correspondía con la versión que aparecía en los archivos de configuración.
- El código de ejemplo no incluía en el archivo de configuración `AndroidManifest.xml` los permisos y características de uso que requería la aplicación para funcionar. De esta forma, cuando intentábamos probar la demo en nuestro móvil, ni siquiera se abría y aparecía un mensaje indicando que se había detenido.

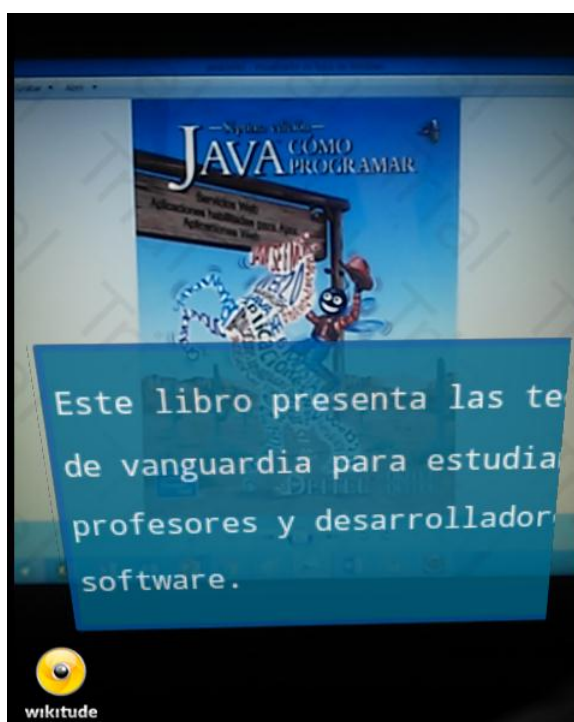


Figura 51: Prueba de reconocimiento en local con Wikitude

- Prueba de reconocimiento en la nube de imágenes

Investigando para conseguir una prueba en local con la versión 5 del SDK de Wikitude, encontramos los *samples* de Wikitude. Se trata de ejemplos sencillos que muestran casos de uso de lo que se puede hacer con Wikitude. Al descargar la última versión del SDK en la web de Wikitude, hay una carpeta que se llama “Examples”. Dentro de esa carpeta hay dos proyectos que se pueden importar en Android Studio. El que nosotros utilizamos en posteriores pruebas fue “SDKExamples”.

Una vez encontramos los *samples* de Wikitude, los tomamos de ejemplo para implementar nuestra primera prueba sencilla en la nube. En esta primera prueba en la nube partimos del ejemplo *SDKExamples > Cloud Recognition > Continuous Recognition vs On-Click*. Las principales modificaciones con respecto al ejemplo de los *samples* fueron:

- Nuestra clase `MainActivity` se basa en la de los ejemplos de Desarrollo Libre, reemplazando la clase `ArchitectConfig` del `ArchitectView` por `StartupConfiguration` para actualizar el código a la versión 5 del SDK.
- Nuestro `AndroidManifest.xml` se basa en el de los ejemplos de Desarrollo Libre, incluyendo los permisos y características de uso necesarios.
- Modificamos el archivo de `continuousrecognitionvsonclick.js` de tal forma que al reconocer una imagen se superponga otra imagen indicando éxito.
- Creamos un proyecto con el *Target Manager* de Wikitude en el que incluimos las imágenes que reconocería nuestra aplicación y añadimos el *Client Token* y el *Target Collection Id* en nuestro código.
- Obtuvimos una licencia de prueba gratuita y la incluimos también en nuestro código.



4.4.2 Pruebas de servicios web REST con Jersey

En primer lugar, instalamos las herramientas necesarias y configuramos el entorno:

- Ya disponíamos de Eclipse en nuestros equipos, pero instalamos el *plugin* WTP para poder crear Dinamic Web Projects.
- Instalamos Tomcat y lo configuramos para poder utilizarlo desde Eclipse. Para ello seguimos un tutorial de Youtube (41).

El siguiente paso fue aprender cómo utilizar Maven para gestionar y construir un proyecto de Java. Esto fue así debido a que la mayoría de tutoriales que encontramos para comenzar con los servicios web utilizaban esta herramienta.

Para entender Maven realizamos un tutorial de *javaHispano* (42). Este tutorial incluye: instalación de Maven, creación y compilación de proyectos Maven, agregación de dependencias, importación de proyectos en Eclipse y ejecución de aplicaciones web con el *plugin* de Jetty para Maven, todo ello desde la consola de comandos.

La primera prueba de servicios web REST que realizamos fue siguiendo un ejemplo sencillo de *Mykong* (43). En él se muestra cómo implementar una especie de “Hola Mundo” en una aplicación web REST con Jersey. El ejemplo parte de una estructura estándar de proyecto web de Maven. Una vez tuvimos esta sencilla aplicación web REST funcionando, ampliamos su funcionalidad implementando la posibilidad de realizar una suma de dos operandos.

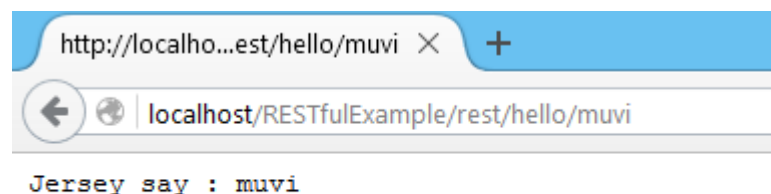


Figura 53: Prueba de saludo

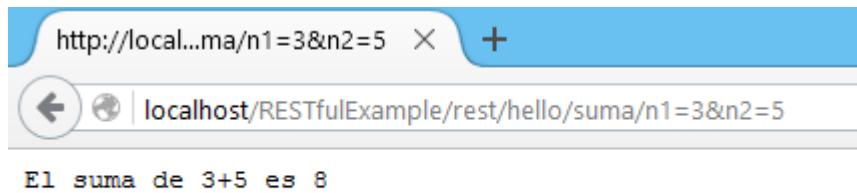


Figura 54: Prueba de suma

4.4.3 Pruebas con Hibernate/JPA

En primer lugar, instalamos las herramientas necesarias y configuramos el entorno:

- Instalamos MySQL a través del MySQL Installer. Incluye MySQL Workbench

Antes de estas pruebas únicamente conocíamos el *framework* de mapeo objeto/relacional EclipseLink. En Hibernate hay dos formas de realizar un mapeo objeto/relacional: a través de archivos de mapeo en XML y usando anotaciones. En un principio elegimos esta última simplemente porque es la que más se parecía a la que habíamos usado anteriormente.

Como primera toma de contacto con Hibernate realizamos un tutorial para aprender a persistir objetos simples usando anotaciones (44). La única diferencia es que nosotros utilizamos Maven para gestionar las dependencias. Algo que tuvimos en cuenta a partir de la realización de este tutorial es la propiedad `hbm2ddl.auto` del archivo de configuración `hibernate.cfg.xml`. En función del valor que tome esta propiedad nos permite: crear las tablas (`create`), actualizar las tablas si hay modificaciones (`update`), y borrar las tablas existentes y crearlas de nuevo (`create-drop`).

Una vez terminamos este primer tutorial nos dimos cuenta de que no usa JPA, únicamente usa sus anotaciones. Esto es así porque no utiliza el *proveedor de persistencia* de JPA, sino que de ese trabajo se encarga Hibernate. Para modificar este primer ejemplo de prueba con Hibernate de tal forma que sí utilizara JPA seguimos otro tutorial (45).

Cuando ya teníamos nuestra pequeña aplicación de ejemplo usando Hibernate y JPA, el siguiente paso fue crear a partir de ella una aplicación web implementando servicios REST con Jersey. Esto nos supuso varios problemas relacionados con la gestión de dependencias. Fue entonces cuando comenzamos a utilizar Maven integrado con Eclipse.

4.4.4 Pruebas de los servicios de redes sociales

Twitter

El objetivo de esta prueba ha sido implementar un servicio que postee en Twitter un mensaje desde nuestra aplicación sobre una película previamente escaneada. Se ha utilizado la librería de Java Twitter4J, que permite manejar la API de Twitter de forma sencilla. Para ello consultamos diferentes fuentes de información y tutoriales (46).

Para poder utilizar la API de Twitter ha sido necesario crearnos una cuenta en Twitter como desarrolladores y, una vez dentro, crear una nueva aplicación para conseguir los *tokens* de acceso que son necesarios para conectarse a Twitter. Para realizar esta configuración inicial hemos seguido un tutorial sobre Twitter4J del blog *Un poco de Java* (47).

La implementación de esta prueba realiza la autorización de Twitter a través de PIN. No es la forma idónea para hacerlo porque el usuario tiene que introducir manualmente en nuestra aplicación el PIN que le facilita Twitter una vez la autorice.



Figura 55: Ejemplo de autorización de Twitter

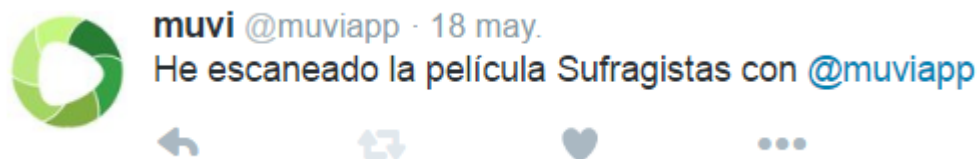


Figura 56: Ejemplo de tweet hecho desde la app

Una opción mejor para implementar este servicio es a través de una URL de vuelta a la app (*callback*), pero en su momento pospusimos su implementación. Para cuando fue retomada, encontramos una forma más sencilla de implementar esta funcionalidad sin la necesidad de un servicio.

Como la RA está realizada mediante tecnologías web, utilizamos la misma técnica que Twitter explica en su documentación (48) para añadir un botón de “Twittear” a una página web. Para ello, implementamos un botón en la RA que al ser pulsado abre Twitter o el navegador y muestra un *tweet* para ser enviado. Se realizó mediante un enlace proporcionado por Twitter en el que puede añadirse el texto que queremos que se envíe desde el usuario que está registrado en Twitter en ese momento. El enlace es el siguiente:

<https://twitter.com/intent/tweet?text=Hello%20world>

Facebook

Con esta prueba hemos pretendido implementar una función en nuestra aplicación que permita publicar en el muro de Facebook una película escaneada con nuestra aplicación. Se ha utilizado, como en el caso de Twitter, una API más sencilla que la oficial de Facebook, Facebook4J.

Siguiendo un tutorial del blog *Un Poco de Java* (49), lo primero que se hemos hecho ha sido crear una cuenta de Facebook y registrarnos como desarrolladores en Facebook.

El siguiente paso ha sido crear una nueva aplicación de Facebook que permite publicar en el muro y, por último, en el panel de control para desarrolladores de Facebook, generar el *token* necesario con los permisos de publicación.

Con todo lo anterior se consiguió la publicación mostrada en la siguiente imagen:



Figura 57: Ejemplo de publicación en Facebook

El problema que se tuvo a continuación fue conseguir los *tokens* que contienen los permisos de publicación de una cuenta ajena a la nuestra. Debido a una reciente actualización de la API de Facebook esos tokens solo pueden obtenerse usando la API para Android, que permite implementar lo necesario para hacer login con una cuenta de Facebook válida y solicitar permisos de publicación mediante un botón, tal y como se puede comprobar en esta otra aplicación de navegación:



Figura 58: Ejemplo de botón de Facebook

Como alternativa a esa opción se ha probado a utilizar directamente una dirección de Facebook que permite publicar webs en el muro, como se puede comprobar en la imagen.



Figura 59: Segundo ejemplo de publicación en Facebook

Al final se descartó la funcionalidad de compartir en Facebook, ya que no nos permitía personalizar la publicación, siendo sólo posible compartir enlaces que no hacen referencia alguna a nuestra aplicación.

4.4.5 Pruebas de geolocalización

GPS

Con esta prueba desarrollamos una forma alternativa de representar a la gente cercana, a la usada en los Beacons, debido a la existencia de algunos problemas presentados durante su desarrollo. Las pruebas de la geolocalización constan de dos partes.

En primer lugar, conseguir los puntos de latitud y longitud, mediante el uso de llamadas a los servicios GPS, construyendo un servicio completo en nuestra aplicación. Durante su realización se presentaron problemas de permisos debido a que la nueva API de Android (SDK 23 de Android), presenta un cambio en los permisos, donde antes los usuarios se despreocupaban de otorgar algunos permisos llamados de riesgo, como el uso de tu localización o información personal. Ahora son los usuarios los que tienen que dar permiso para el uso de algunos recursos de las aplicaciones, delegando de esta forma al programador la llamada explícita para la aceptación del permiso.

En segundo lugar, se realizaron pruebas de cálculo de distancias. Primeramente, dejamos el cálculo a la aplicación mediante una función, ofreciendo una distancia de error desde el primer punto calculado a los siguientes puntos que se calculan. Seguidamente nos dimos cuenta que podíamos hacer este cálculo directamente en el servidor, mediante MongoDB, que ofrece una función específica que calcula la distancia.

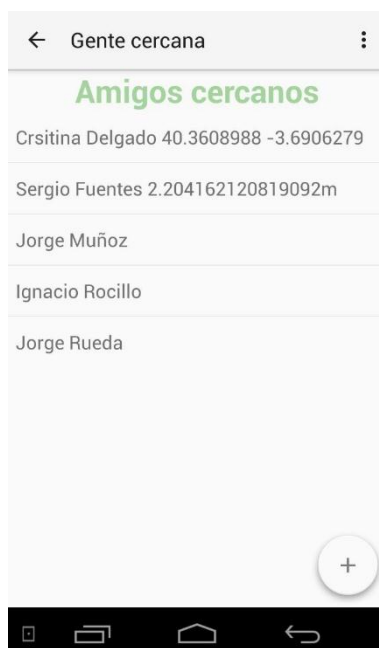


Figura 60: Prueba de geolocalización

Capítulo 5: EVALUACIÓN DE USUARIOS

Con una versión de la aplicación funcional y prácticamente estable, comenzamos a realizar las evaluaciones con usuarios para estudiar la usabilidad de nuestra aplicación y comprobar si hay posibles problemas a la hora de ser manejada por gente ajena al proyecto.

Exponemos el plan de evaluación, donde detallamos los objetivos de la evaluación, las tareas que los usuarios deben realizar durante la evaluación, las preguntas que deben responder y cuál será nuestro trabajo en cada evaluación.

También mostramos las opiniones de los usuarios a nuestra aplicación una vez realizada la evaluación, contando cuántas personas la hicieron, los errores que encontramos, opiniones y críticas hacia la aplicación y conclusiones que tomamos.

5.1 Plan de evaluación

Este plan de evaluación se basa en el que realizaron en el TFG de RACMA, una aplicación de RA para museos (50)

5.1.1 ¿Qué estamos evaluando?

Proyecto: MUVI APP

Premisa: Un grupo de amigos va al cine, pero no se ponen de acuerdo en qué película ir a ver, y necesitan información de la película y qué película se les recomienda. Además, un usuario ve un cartel de una película y quiere ver todo lo relacionado con ella.

Objetivo de la aplicación: Ayudar a grupos de gente que van al cine a decidir qué película ver, mostrando mediante una interfaz de RA y apuntando a un cartel, información de dicha película, tal como tráiler, sinopsis y puntuación media en medios internacionales de opinión, y valoración que su grupo de amigos daría. Además, se permite añadir la película a una lista de deseos para poderla ver en otro momento.

5.1.2 Propósito de la evaluación

Una vez terminada la etapa de desarrollo, se desea comprobar el impacto del uso de la RA en el usuario, así como la usabilidad de la aplicación.

5.1.3 Objetivos generales

Deseamos evaluar las principales características de la aplicación:

- Recabar la opinión del usuario sobre el uso de la aplicación, especialmente sobre la interfaz de RA.
- Comprobar la facilidad de uso de la aplicación: si la interfaz es sencilla, cómoda e intuitiva.

5.1.4 Preguntas de investigación

- ¿Se puede usar *MUVI* de manera correcta?
- ¿Es confusa la interfaz?
- ¿Los botones son claros y están bien colocados?
- ¿El usuario encuentra de manera fácil lo que busca?
- ¿La aplicación muestra la información necesaria al usuario?
- ¿Hay alguna función de la aplicación que se echa en falta?

5.1.5 Requisitos de los participantes

Las personas a las que entrevistaremos para las evaluaciones serán familiares, amigos y/o conocidos, preferiblemente con rangos de edades diferentes y con niveles de estudios diferentes.

5.1.6 Diseño experimental

- La evaluación se realizará entre los días 20 y 22 de mayo de 2016
- Se realizarán de 5 a 10 evaluaciones, teniendo cuidado de evaluar a gente con rangos de edades y niveles diferentes.
- Duración aproximada de cada evaluación individual: 20 minutos.
- Duración total de la evaluación: aproximadamente 40 minutos por cada miembro del grupo, haciendo un total de unas 5 horas para todos los usuarios.

El plan a realizar con el usuario es:

1. Realizar un cuestionario para saber datos demográficos del usuario y el conocimiento del usuario sobre RA en un primer momento.
2. Introducir al usuario sobre qué es la RA. También explicarle en qué consiste nuestra aplicación, sin entrar demasiado en detalles.
3. Proporcionarle una lista de tareas al usuario y pedirle que las realice. Debe realizar las tareas sin nuestra ayuda, a no ser que se quede atascado sin saber cómo seguir.

Mientras tanto, iremos tomando nota de cómo el usuario realiza cada una de las tareas.

El usuario, a su vez, deberá ir expresando en voz alta sus impresiones.

4. Pequeño diálogo con el usuario, preguntándole qué es lo que más le ha gustado y si tiene alguna queja en concreto.
5. Realización de un cuestionario sobre la facilidad de uso de nuestra aplicación.
6. Recopilación de datos:
 - a. A partir de la encuesta, obtendremos datos cuantitativos sobre la experiencia del usuario.
 - b. Con la información tomada durante la observación, obtendremos datos cualitativos que nos permitirán detectar cuáles son los errores más comunes que han cometido los usuarios para su posterior análisis.

5.1.7 Tareas a realizar

Esta lista de tareas está relacionada con nuestra aplicación.

1. Regístrate en la aplicación.
2. Inicia sesión en la aplicación.
3. Entra en la interfaz de RA.
4. Reconoce una de las películas.
5. Muestra la información sobre la película.
6. Reproduce el tráiler de la película.
7. Pausa el tráiler de la película.
8. Deje de mostrar el tráiler de la película.
9. Añade la película a "Me Gusta".
10. Realiza una valoración de la película.
11. Vuelve a la pantalla de inicio.
12. Muestra la lista de películas que te gustan.
13. Muestra la información de una de las películas favoritas.
14. Vuelve a la lista de favoritos.
15. Vuelve a la pantalla de inicio.
16. Muestra la gente cercana.
17. Vuelve a la pantalla de inicio.

5.1.8 Entorno y herramientas empleadas

No es obligatorio ningún entorno específico para la realización de las pruebas, únicamente valdrá con que sea un espacio cómodo donde poder realizarlas sin problemas ni interrupciones.

Las herramientas que utilizaremos serán un teléfono móvil y fotocopias o imágenes de los carteles de película que reconoce la aplicación. No haremos uso del dispositivo Beacon ni de la recomendación a grupos debido a que su funcionalidad está diseñada para grupos y nuestras pruebas se realizarán individualmente.

5.1.9 Obtención de *feedback* de los participantes

Las fuentes de datos serán las siguientes:

1. Cuestionario inicial para conocer el contexto del usuario antes de realizar la evaluación.
2. Reunión posterior a la evaluación con el participante.
3. Cuestionario SUS traducida por Marta Caro y David Hernández (50), que medirá el grado de usabilidad. Las preguntas se responden a través de una valoración del 1 al 5, en la que 1 es totalmente en desacuerdo y 5 totalmente de acuerdo.

5.1.10 Tareas del moderador

Las tareas que deberá realizar el moderador a la hora de la evaluación serán:

- La primera tarea del moderador será explicar al usuario en qué consisten tanto la aplicación como la RA.
- Pedirá al usuario que realice las tareas anteriormente expuestas, y anotará y observará todo aquello que considere relevante para el estudio. Importante detectar los problemas que el usuario encuentre. A lo largo de estas tareas, el moderador no ayudará al usuario a menos que sea totalmente necesario.
- Pedirá al usuario que comente en todo momento sus impresiones de la aplicación y de la tarea asignada.
- Para terminar, proporcionará al usuario las encuestas para la recolección de datos para que las realice.

5.1.11 Descripción de la metodología de análisis de datos

1. Los cuestionarios a los usuarios se realizarán mediante la herramienta Google Forms, ya que se realizarán en espacios conectados a Internet.
2. Los moderadores anotarán en papel todos los apuntes que crean convenientes en un documento impreso con el nombre de cada tarea y un espacio en el que añadir texto. A la hora de agrupar todos los formularios, se escanearán (o en caso de ser imposible el escaneado, se realizará una foto) y se guardarán en la carpeta de Google Drive preparada para la tarea.

3. Con las notas y los valores recogidos en los cuestionarios, se comprobará cuáles son las tareas que más tiempo tardan en realizarse y se estudiará si la implementación es la correcta.
4. Se pondrán en común todas las conclusiones y se realizará un informe con todo los errores y posibles mejoras que puedan realizarse.

5.2 Evaluación

Durante la evaluación comprendida entre los días 21 y 22 de mayo, efectuamos 7 evaluaciones de 20-30 minutos cada una. El rango de edades de los usuarios fue el siguiente:

- Entre 18 y 29 años: 4
- Entre 30 y 45 años: 1
- Mayores de 46 años: 2

Además, hemos intentado que tengan una profesión o estudios diferentes para poder tener un mayor rango de opiniones. Los usuarios eran: estudiante de Bachillerato, estudiante de Geología, estudiante de Relaciones Laborales, electromecánico, Ingeniero Informático, Ingeniero de Telecomunicaciones y pensionista.

5.2.1 Observaciones y opiniones de los usuarios

Empezamos la evaluación preguntando al usuario cuál es su experiencia sobre la RA. Los resultados fueron los siguientes:

- 6 de los 7 evaluados no poseían experiencia previa en aplicaciones de RA.
- Sólo un evaluado tenía experiencia previa en RA, en videojuegos y en museos.

En la realización de tareas, los principales problemas que encontraron los usuarios por cada tarea fueron los siguientes:

- Regístrate en la aplicación: algunos usuarios no encontraron el botón de registro, ya que el teclado sale automáticamente y tapa el botón. Otro usuario detectó falta de consistencia de los idiomas entre “Registrarse” y “Login”.
- Entra en la interfaz de RA: un usuario se extrañó de la petición de encender el *bluetooth*. Se le explicó que el motivo era la búsqueda de los usuarios cercanos. Otro usuario dudó como entrar.
- Reconoce una de las películas: todos los usuarios intuitivamente apuntaron a la película, pero algunos se quejaron de que se reinicia el servicio al girar la pantalla.

- Muestra la información sobre la película: no tuvieron problema en encontrar el botón. Sin embargo, algunos usuarios pidieron que se mostrara sobre que puntuación es la valoración máxima y a otros no les gustó que si mueves el móvil se va la captura.
- Reproduce el tráiler de la película: la mayoría no tuvo problemas a la hora de encontrar el botón. Algunos usuarios pidieron poder cambiar el tamaño o se quejaron de la incomodidad de tener el brazo levantado para poder ver el video. Sin embargo, también les gustó el hecho de que el vídeo cargara tan rápido y que se muestre en la RA.
- Pausa el tráiler de la película: un usuario necesitó ayuda del moderador para pausar el video.
- Deje de mostrar el tráiler de la película: algunos usuarios intentaron arrastrar el video para que dejara de mostrarse y otro buscó una X cerca del video.
- Añade la película a tus favoritos: un usuario pidió que se mostrase un mensaje de notificación.
- Realiza una valoración de la película: a todos los usuarios les cuesta encontrar el botón o deducen que es ese por descarte. También se quejan de no poder rectificar una valoración. Otro usuario ve la aplicación como una herramienta para elegir que ver, no cree que nadie vaya a buscar el póster para puntuarla una vez vista la película.
- Muestra la lista de películas que le gustan: todos los usuarios han encontrado problemas a la hora encontrar el menú principal. Piden cambiar el icono del menú porque la flecha parece que saca de la aplicación.
- Vuelve a la lista de favoritos: vuelven con el botón de Android, no de la aplicación.

Con los resultados de las pruebas, sacamos una lista de problemas y soluciones ordenados por orden de prioridad y cómo lo resolveríamos en un trabajo futuro:

1. Mejorar el rendimiento de la aplicación para que no se reinicie el servicio de RA al girar el dispositivo.
2. Cambiar los iconos de algunos botones, como el de Valoración de la RA o el del Menú de Inicio debido a la confusión que causaron a todos los usuarios.
3. Cambiar el inicio de la aplicación para que no salga automáticamente el teclado y no esconda el botón de Registrarse.
4. Permitir cambiar la valoración individual debido a que actualmente no es posible cambiarla.
5. Añadir otra forma de quitar el video de la RA para que sea más intuitivo, como una X o arrastrar el video.
6. Cambiar la puntuación de los medios para que pueda verse sobre qué nota está puntuada la película.

5.2.2 Resultado cuestionario SUS

Al final de la realización de las tareas, se pidió a los usuarios que realizaran el test de usabilidad SUS de la aplicación con el fin de obtener la puntuación SUS y comprobar el nivel de facilidad de uso de nuestra aplicación.

Para obtener la puntuación SUS se siguen los siguientes pasos:

1. Para las preguntas impares se resta uno al valor respondido.
2. Para las pares se resta al valor 5 el valor respondido.
3. Sumamos todos los valores obtenidos tras realizar los pasos 1 y 2.
4. Multiplicamos por 2,5 y obtendremos la puntuación SUS.

Durante la evaluación se ha obtenido una **valoración media de 80,7**, una valoración por encima de los 68, resultado considerado satisfactorio, por lo que se puede afirmar que nuestra aplicación ha obtenido una buena puntuación en esta encuesta. Además, la encuesta SUS organiza las puntuaciones utilizando una escala de valores, en la que una F es un suspenso de la aplicación y una A es la puntuación máxima a obtener. Para conseguir una A es necesario superar un 80 de puntuación, y la nuestra es de 80,7, por lo que se puede concluir que la usabilidad de MUVI es sobresaliente.

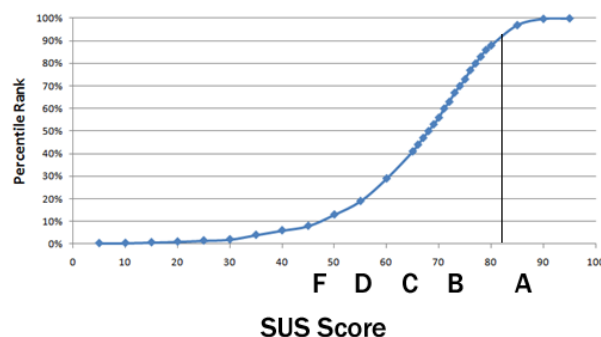


Figura 61: Resultados posibles de SUS

Conclusiones y sugerencias

Una de las quejas más habituales fue la postura a la hora de realizar la prueba en la RA, y una propuesta que nos dieron fue que no fuera obligatorio estar constantemente apuntando a la película, lo que llevaría a un cambio muy grande en la estructura de la RA.

Otro aspecto muy importante del que los usuarios se quejaron fue del botón de valoraciones en la RA, que no queda nada claro y es difícil de encontrar, por lo que habría que cambiar ese icono por otro más intuitivo.

Tampoco gustó el icono del menú principal, ya que mucha gente no sabía qué hacía ese icono o pensaba que sacaba de la aplicación, por lo que sería conveniente cambiarlo también.

Entre las funcionalidades que añadirían ellos a la aplicación, nos dijeron que la RA podría llevar a la compra de entradas en alguna web y qué considerarían de mucha ayuda que mostrara opiniones de los usuarios de la película, no sólo la valoración media de webs importantes.

Sin embargo, hemos comprobado en las encuestas que, a pesar de los pequeños fallos que encontraron en la aplicación, todos los usuarios han expresado su aprobación a la aplicación y algunos han quedado impresionados con ella. A todos les ha gustado e incluso han afirmado que estarían dispuestos a pagar por ella en el caso de que saliera a la venta. Han opinado que es una aplicación realmente útil a la hora de ir al cine y que la recomendarían.

Capítulo 6: CONCLUSIONES

6.1 Conclusiones

Cuando se inició el desarrollo de este Proyecto, se estableció unos objetivos principales y una serie de objetivos parciales cuya consecución determinaría el resultado del Proyecto.

En los próximos puntos se detallarán todos los objetivos principales que se han logrado:

- Se ha definido y entendido la tecnología de la realidad aumentada mediante la búsqueda de información y la realización de diversas pruebas, integrándola desde el primer prototipo de nuestra aplicación, añadiendo nuevas funcionalidades y resolviendo los problemas surgidos durante la etapa de desarrollo.
- Se ha investigado y aplicado el funcionamiento de las tecnologías de los Beacons durante la etapa de desarrollo para registrar los grupos de amigos presentes en el cine.
- Se ha investigado e implementado el uso de sistemas de recomendación para ofrecer a los amigos una valoración grupal de las películas. Para abordar este desafío se utilizó Mahout, que se ajustaba a los requisitos de la aplicación, proporcionándonos la posibilidad de realizar primero, recomendaciones individuales y, después, de grupo.
- Se ha diseñado una arquitectura completa y funcional que abarca el uso de todas las tecnologías anteriormente descritas.
- Se ha realizado una evaluación con los usuarios, que nos ha permitido comprobar problemas de usabilidad y detectar algunos fallos de la aplicación, que nosotros como desarrolladores no habíamos contemplado en un primer momento y nos ha permitido observar la importancia que tienen en un proceso de desarrollo las evaluaciones con los usuarios.

En los próximos puntos se explicarán los diferentes objetivos parciales, necesarios para la satisfacción de los objetivos principales:

- Se ha aprendido a programar en Android, conociendo su sintaxis y características, lo que nos ha permitido desarrollar y establecer la lógica de la aplicación.
- Se ha estudiado y comprendido cómo es la arquitectura de cliente-servidor y el funcionamiento de los servicios web, que ha permitido establecer una comunicación entre servidor y cliente.
- Se ha logrado manejar con soltura el entorno de desarrollo Android Studio, lo que ha permitido realizar todo el trabajo de programación de la parte cliente, pruebas de la aplicación, detección de errores en el código, interfaz de usuario, etc.

- Se han estudiado los diferentes entornos de desarrollo de RA disponibles en Unity y en Android, consiguiendo tener una visión global de su funcionamiento y características que nos proporcionan.
- De los entornos anteriores, se ha comprendido y estudiado el funcionamiento de dos en particular, Vuforia y Wikitude. Posteriormente, elegimos Wikitude para aprender su arquitectura y su funcionamiento en Android.
- Se han aprendido a usar los sistemas de localización en Android, tanto el sistema GPS tradicional como el sistema de localización en interiores mediante dispositivos Beacon. Se han estudiado los diferentes comportamientos y aprendido a usar las herramientas que nos ofrecía, en particular, Kontakt.io en el desarrollo de la localización de personas mediante dispositivos Beacon.
- Se ha logrado mantener estable un servidor, proporcionado por la facultad, descubriendo los principales problemas y las posibles soluciones para su reparación.
- Nos hemos instruido en el uso de sistemas de recomendación, en especial, Mahout, detectando los problemas principales que tienen estos sistemas, como el arranque en frío.
- Se ha obtenido una sinergia entre las tecnologías de RA, el sistema de localización y el sistema de recomendación, permitiéndonos realizar nuestro proyecto con éxito.

Durante la realización del proyecto han ido apareciendo problemas o dificultades con el desarrollo de la aplicación.

Las dificultades de reconocer imágenes con una tonalidad oscura, la utilización de cámaras de baja calidad o con unas condiciones desfavorables de iluminación. Todo esto impide que muchas veces la RA no tenga un correcto funcionamiento. Esta tecnología todavía necesita un poco de maduración para solventar los problemas encontrados.

También se encontró el problema de detectar una señal regular del sistema de localización interno a lo largo del tiempo e incompatibilidades con versiones antiguas de Android. Todo ello nos plantea que la tecnología aún tiene que mejorar en la efectividad de transmisión de señales, además que gran parte de la población disponga de un dispositivo compatible.

La mayor adversidad que hemos encontrado al utilizar los sistemas de recomendación ha sido una herramienta que no estaba actualizada para funcionar con versiones de Java utilizadas en la parte de servidor. Por consiguiente, tuvimos que encontrar otra herramienta que tuviera un correcto funcionamiento.

Resumiendo, se han satisfecho los objetivos principales que tenía el proyecto y que se marcaron en un inicio, ya que se ha logrado tener una versión estable de nuestra aplicación, con todas las funcionalidades requeridas.

Este Proyecto tendrá unas implicaciones empresariales en el sector del ocio, donde encuadramos el uso de nuestra aplicación. Nuestra aplicación ofrecerá una novedosa forma de afrontar los problemas comunes de los usuarios, dándoles la posibilidad de obtener la información de una forma sencilla y atractiva. También, recomendándoles películas para ellos y su grupo de amigos.

Finalmente, concluir que el desarrollo de MUVI ha sido muy satisfactorio y gratificante para nosotros, tanto a nivel académico como personal. Hemos aprendido una gran cantidad de cosas durante todo el proyecto, ofreciéndonos una visión mucho más amplia de las tecnologías que se usan en el mundo laboral. Esperamos que nuestra aplicación sea del agrado de todos los usuarios que la utilicen y la disfruten tanto como nosotros hemos disfrutado creándola.

6.2 Conclusions

When the development of this project began, several major objectives and a series of less important objectives were established. The achievement of these objectives would determine the outcome of the project.

The next points describe the major objectives that have been achieved:

- It has been defined and understood the augmented reality technology by seeking information and conducting various practical tests, integrating it from the first prototype of our application, adding new features and solving the problems encountered during the development stage.
- It has been explained and applied the Beacons technologies during the development stage to record groups of friends that are near to the Beacon.
- It has been researched and implemented a recommendation system to offer friends a group evaluation of films. To address this challenge, we used Mahout, which fits the application requirements, providing the possibility to make individual recommendations, first, and then group based recommendations.
- It has been designed a complete and functional architecture that encompasses the use of all the technologies described above.
- It has carried out an evaluation with users, which has allowed us to test usability problems and detect some application failures that we as developers had not contemplated at first. This has allowed us to observe the importance of a development process evaluations with users.

After explaining the main objectives that have been met, in the next points the different partial objectives will be explained:

- We have learnt to program Android applications, knowing its native syntax and features, what has allowed us to develop the logic of the application.
- We've studied the client-server architecture and how Web services work.
- We've learnt to use Android Studio, the programming environment. This has allowed all the client side programming, the testing of the application, error detection, programming of the user interfaces etc.
- We have studied the different development environments for Augmented Reality in Android and Unity, so we got to know an overview of its performance and features that they provide.
- From the all the available environments, we studied the functionality of two in particular, Vuforia and Wikitude. Later we chose Wikitude for the development of the augmented reality.
- We have learnt to use the Android location systems. The traditional GPS system as well as the Beacons, the indoor location system. Studying the different behaviors and learning how to use the tools that they offered us. Particularly Kontakt.io, that we used in the development of the location of people using the Beacon.
- We managed to maintain a stable server, discovering the main problems and possible fixes for them.
- We have been instructed in the use of recommendation systems, especially Mahout. Detecting the main problems with these systems, such as the cold start. Mahout has been used for user group based recommendations.
- It has obtained a synergy between the technologies of Augmented Reality, the location system and recommendation system, that has allowed us to achieve our project successfully.

During the project they have appeared some problems or difficulties related with the development of the application. For example, the difficulties of recognizing images with a dark hue, the use of low-quality cameras or unfavorable lighting conditions. All these problems make the Augmented Reality not to have a proper functioning. This technology still needs some maturing in order to solve the encountered problems.

Also, we found the problem of detecting a regular signal of the location system over time as well as incompatibilities with older versions of Android. This make us to think that the technology still needs to improve the effectiveness of signal transmission. Moreover, most people would need to have a supported device.

The greatest adversity we found when using recommendation systems, it's been a tool that was not updated to work with versions of Java that we used on the server side. Therefore, we had to find another tool that had a proper updated version and functionality.

Summing up, we have satisfied the main objectives of the project that was marked in the beginning. Since we have managed to have a stable version of our application, with all of the required functionalities.

This project will have some implications in the leisure sector, where we have framed the use of the application. Our application will provide a novel way to tackle common user's problems, enabling them to get the information in a simple and attractive way. Also, the app will recommend movies for them and their group of friends.

Finally, we conclude that the development of MUVI has been very satisfying and rewarding for us, academically and personally. We have learned a lot of things throughout the project. We hope that our application will please all users and that they will enjoy it as much as we enjoyed creating it.

Capítulo 7: TRABAJO FUTURO

El objetivo de nuestro proyecto ha sido desarrollar una aplicación. Para ello hemos tenido que aprender a usar diferentes tecnologías. Tuvimos que adaptarnos al entorno de desarrollo de Wikitude y de Android, y el uso de un servidor ajeno a nosotros de la facultad. Por lo tanto, tuvimos algunos objetivos propuestos en un principio que no hemos podido abarcar, debido al tiempo y a la falta de familiarización de la arquitectura que estábamos usando.

Aun así, hemos podido terminar un proyecto de una aplicación móvil completa y un servidor completo para su uso funcional, pudiendo así cometer varias ampliaciones en un futuro, para dotar a nuestra aplicación de mayor funcionalidad.

En los siguientes puntos, se repasarán posibles cambios y nuevas funcionalidades, que no se tuvieron anteriormente en consideración debido a ser de baja prioridad para el funcionamiento de la aplicación, y resultados obtenidos de la evaluación con los usuarios.

1. Añadir la posibilidad de hacer planes con tus amigos para ir a ver una película que en conjunto guste a todos.
2. Incorporar más idiomas además del castellano, para usarse en todo el mundo.
3. Añadir una ventana de amigos en la interfaz, donde podamos en todo momento invitar a los diferentes amigos a ir a ver una película y también visualizar los gustos y valoraciones de los diferentes amigos.
4. Añadir foto de perfil a la cuenta y adaptar la interfaz para este uso.
5. Añadir de forma dinámica el uso de videos del tráiler de la película, para así no abultar el peso de nuestra aplicación y poderla subir a Play Store.
6. Añadir de forma dinámica la cartelera de cada semana en nuestra base de datos y en Wikitude.
7. Añadir la posibilidad de poder comentar en las películas, para así darle un sentido de red social para ir al cine.
8. Estudiar y mejorar la estabilidad de la aplicación, ya que hemos presenciado que el servidor es inestable y se interrumpen los diferentes servicios para que la aplicación funcione correctamente.
9. Mejorar el rendimiento de la aplicación para que no se reinicie el servicio de RA al girar el dispositivo.
10. Cambiar los iconos de algunos botones, como el de Valoración de la RA o el del Menú de Inicio debido a la confusión que causaron a todos los usuarios.
11. Cambiar el inicio de la aplicación para que no salga automáticamente el teclado y no esconda el botón de Registrarse.
12. Permitir cambiar la valoración individual.

13. Añadir otra forma de quitar el video de la RA para que sea más intuitivo, como una X o arrastrar el video.
14. Cambiar la puntuación de los medios para que pueda verse sobre qué nota está puntuada la película.

Capítulo 8: ORGANIZACIÓN DEL TRABAJO

En la etapa de investigación y pruebas con tecnologías, no ha existido división del trabajo como tal. Cada uno de los miembros investigaba lo que consideraba conveniente sobre la tecnología objetivo en cada momento.

En la etapa de desarrollo, nos hemos dividido en dos grupos. Uno se ha centrado en el desarrollo del *front-end* y lo han formado Ignacio Rocillo, Jorge Muñoz y Sergio Fuentes. El otro se ha centrado en el desarrollo del *back-end* y lo han formado Cristina Delgado y Jorge Rueda.

En la etapa de realización de la memoria, nos hemos dividido el trabajo de tal forma que cada uno de nosotros se ha enfocado en escribir o documentar sobre aquello con lo que ha estado más en contacto.

A continuación, se muestra la cronología del proyecto:

	19-10-02-11	02-11-11	16-11-11	30-11-12	14-12-11-01 (Navidad)	11-01-25-01	15-02-29-02	29-02-16-03	16-03-06-04 (S. Santa)	06-04-20-04	20-04-04-05	04-05-18-05	18-05-23-05	23-05-06-06
ETAPA DE INVESTIGACIÓN														
Investigación tecnologías RA														
Definición casos de uso														
Prototipado de las IU														
Pruebas RA														
Investigación servicios REST														
Definición servicios API														
Pruebas tecnologías servicios														
Pruebas geolocalización GPS [1]														
Investigación recomendación [2]														
ETAPA DE DESARROLLO														

Interfaz RA														
Servicios de usuarios														
Servicios de películas														
Servicios de deseos														
Servicios de redes sociales														
Interfaz de Registro														
Interfaz de Login														
Diseño y maquetación de logo														
Servicios de valoraciones														
Prototipo 1 Interfaz de Inicio														
Servicios de <i>beacons</i>														
Desarrollo <i>beacons</i> en cliente														
Servidor FDI [3]														
Prototipo 2 Interfaz de Inicio														
Interfaz de Ficha de Película														
Interfaz de Me Gusta														
Cambios en servicios														
Mejoras Interfaz RA														
Mejoras interfaces Android														
Interfaz de Gente Cercana														
Servicios de recomendación														
Demo estable [4]														
ETAPA DE MEMORIA														
Documentación servicios API [5]														

Documentación pruebas [6]														
Estructura de la memoria														
Versión 1 de la Memoria														
Evaluación de usuarios														
Versión 2 de la Memoria														
Versión 3 de la Memoria														

La etapa de investigación corresponde con la primera etapa del proyecto. Sin embargo, ciertas tareas de investigación se han realizado durante etapas posteriores.

En el caso de las pruebas de geolocalización GPS [1], no estaba previsto su uso en un principio, sino que surgió como una alternativa a la localización con dispositivos Beacons cuando tuvimos ciertos problemas con su desarrollo.

En el caso de la investigación sobre sistemas de recomendación [2], en un primer momento íbamos a utilizar el proyecto jCOLIBRI para el desarrollo de nuestro sistema de recomendación, pero debido a la incompatibilidad de las versiones de Java entre este proyecto y nuestra API de servicios fue necesario buscar alternativas.

La etapa de desarrollo corresponde con la segunda etapa del proyecto. Sin embargo, ciertas tareas de desarrollo se han alargado hasta el final del proyecto.

En el caso del despliegue de la aplicación en el servidor proporcionado por la Facultad de Informática [3], este servidor forma parte de un proyecto piloto, del cual hemos sido un grupo experimental. Esto nos ha conllevado una serie de inconvenientes, que se describen en el capítulo “Servidor” dentro del apartado “Implementación” de este documento. Es por esta razón que esta tarea se ha alargado hasta el final del proyecto. La tarea de conseguir una demo estable [4] de nuestra aplicación está relacionada también con los problemas surgidos con este servidor.

La etapa de memoria corresponde con la última etapa del proyecto. Sin embargo, ciertas tareas relacionadas con la memoria se han realizado con anterioridad.

En el caso de la documentación de los servicios de la API [5], esta documentación se ha realizado con anterioridad para utilizarla como referencia, tanto en el *back-end*, como en el *front-end*, durante el desarrollo.

En el caso de la documentación de las pruebas [6], estas pruebas se han documentado a medida que se han ido haciendo o al poco tiempo para evitar el olvido del proceso de realización de las mismas.

8.1 Modelo de desarrollo

Durante el desarrollo de la aplicación utilizamos el *framework* de desarrollo ágil de software Scrum. Se basa en realizar continuamente pequeñas entregas de productos tangibles. Cada iteración (sprint) finaliza con la entrega de una parte operativa del producto (incremento). La duración de cada sprint tiene una duración típica de dos a cuatro semanas.

Siguiendo Scrum, al comenzar la etapa de desarrollo, creamos una lista de tareas priorizadas. Cada uno de nuestros sprints duraba dos semanas. Estos sprints comenzaban y terminaban con una reunión con los tutores del TFG. En cada una de esas reuniones se mostraban las nuevas funcionalidades, se definían las tareas a llevar a cabo en el siguiente sprint y se repartía el trabajo entre los miembros del grupo. Las tareas estaban individualizadas y cada uno de nosotros era responsable de sacar adelante su parte del trabajo.

8.2 Herramientas de comunicación

- Gmail, para la comunicación con los tutores del TFG en momentos puntuales.
- Google Drive, para compartir todo tipo de archivos necesarios para el desarrollo del proyecto (imágenes, documentación, enlaces, código...), tanto entre nosotros, como con los tutores del TFG.
- Telegram, para comunicarnos de forma rápida entre nosotros.

8.3 Herramientas de control de versiones

Como sistema de control de versiones utilizamos Git y nuestro proyecto lo alojamos en GitHub, que es una plataforma de desarrollo colaborativo que utiliza el sistema de control de versiones Git. De esta forma, conseguimos mantener versiones de código estables en un lugar seguro.

Para el uso de Git existen varias alternativas, se puede utilizar desde la consola, la aplicación oficial de GitHub para escritorio u otras. En nuestro caso se han utilizado:

- Aplicación GitHub de escritorio: La aplicación permite usar todas las funcionalidades de Git: *commit*, *push*, etc. Es muy cómoda, ya que permite ver en forma de líneas de tiempo todas las diferentes ramas que tiene cada repositorio, entre otras cosas.
- SourceTree: Es otra aplicación de escritorio, muy parecida a la de GitHub, pero en este caso tiene un control de conflictos más potente. Además, no solo sirve para *GitHub*, también para Bitbucket y otros tipos de control de versiones como *Mercurial*.

8.4 Herramientas de gestión de tareas

Al utilizar una metodología de desarrollo ágil como la de Scrum, hemos necesitado disponer de una gestión del tiempo y de las tareas bastante flexible, cómoda y eficaz. Para ello buscamos entre varias herramientas, pero al final nos decantamos *por Trello*.

Trello es una herramienta de colaboración que organiza los proyectos en tableros. Es decir, que gracias a Trello, se puede saber cuáles son las tareas que se llevan a cabo, quién trabaja en una tarea determinada y cuál es el estado de un proceso.

Es una herramienta muy fácil y visual. Permite llevar un control de todas las tareas del proyecto en forma de tarjetas. Cada tarjeta se organiza en diferentes secciones o *tarjeteros*. Por ejemplo, en nuestro caso, hemos utilizado los *tarjeteros*: “Por hacer”, “En proceso”, “Terminado”.

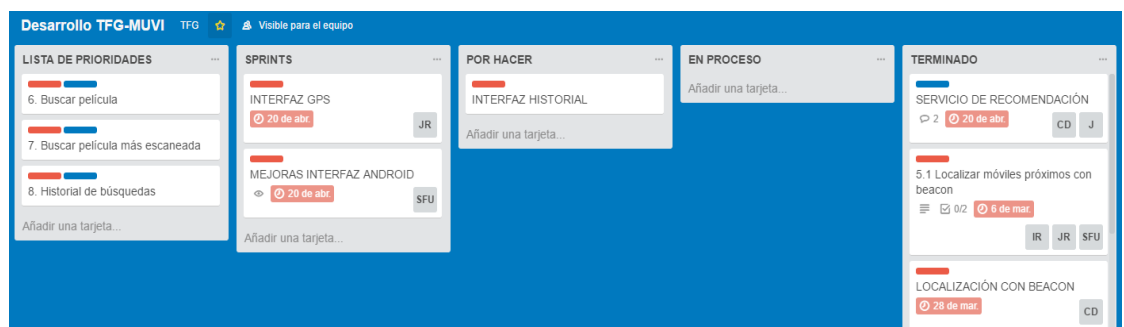


Figura 62: Ejemplo de Trello

Capítulo 9: APORTACIONES AL PROYECTO

9.1 Cristina Delgado Rodríguez

Mi aportación en la etapa de investigación ha consistido en:

- Investigación sobre las distintas tecnologías de RA, con el fin de elegir aquella que más se adaptara a nuestras necesidades.
- Colaboración en la aportación de ideas y posterior definición de la funcionalidad de la aplicación.
- Desarrollo de aplicaciones básicas de prueba de la RA, tanto con Android + Wikitude, como con Unity + Vuforia. El uso de estas tecnologías fue totalmente nuevo. En ninguna de las asignaturas de la carrera había aprendido o utilizado con anterioridad ni Android Studio, ni Unity, y mucho menos algo relacionado con la RA, como es el caso de Vuforia o Wikitude. Tampoco algunos de los lenguajes utilizados (C# para Unity + Vuforia o tecnologías web para Wikitude). El único conocimiento relacionado que tenía en un principio era Java, que es el lenguaje utilizado para el desarrollo en Android. Las aplicaciones que he creado como toma de contacto con ambas tecnologías han sido:
 - Prueba de reconocimiento en local de una imagen, con Unity + Vuforia.
 - Prueba de reconocimiento en local de una imagen, con Android + Wikitude.
 - Prueba de reconocimiento en la nube de imágenes, con Android + Wikitude.
- Investigación en profundidad sobre la creación de servicios de arquitectura REST.
- Colaboración en la definición de los servicios de la API.
- Realización de tutoriales para aprender a utilizar el servidor de aplicaciones, Tomcat, y la herramienta para la gestión y construcción de proyectos Java, Maven.
- Desarrollo de aplicaciones básicas con las distintas tecnologías implicadas en el desarrollo de servicios (Jersey e Hibernate/JPA). El desarrollo de servicios web en mi caso no era nuevo, pero sí el desarrollo de servicios web en Java con Jersey. Por otro lado, sí conocía el estándar de desarrollo JPA, pero no con el *framework* Hibernate. Las aplicaciones de prueba que he creado como toma de contacto con estas tecnologías han sido:
 - Las pruebas de servicios web REST con Jersey
 - Las pruebas con Hibernate/JPA
- Creación de la aplicación básica inicial, integrando las distintas tecnologías implicadas en el desarrollo de los servicios, para el posterior desarrollo de los mismos.
- Investigación sobre el funcionamiento de los dispositivos Beacon y su implementación en Java con el SDK para Android de Konkakt.io

- Investigación sobre el funcionamiento de los sistemas de recomendación y su implementación en Java.

En la etapa de desarrollo:

- Mi principal aportación ha sido en el *back-end*:
 - Desarrollo de la API REST de servicios, junto con Jorge Rueda.
 - Realización de los scripts en Python necesarios.
 - Realización del servicio de prueba de publicación en Twitter.
- Mi aportación en el *front-end* ha consistido en:
 - Desarrollo de la funcionalidad asociada a los dispositivos Beacon.
 - Colaboración en la integración de la aplicación Android.

Mi aportación en la etapa de memoria ha consistido en:

- Creación de la estructura de la memoria.
- Redacción del estado del arte de los sistemas de recomendación (apartado 2.3).
- Realización y explicación del diagrama de arquitectura (en el apartado 4.1).
- Redacción del apartado 4.1.1 Relación entre *front-end* y *back-end*.
- Realización y explicación del diagrama de módulos del *back-end* (en el apartado 4.2).
- Redacción del apartado 4.2.1 Arquitectura de servicios
- Redacción del apartado 4.2.3 Sistema de recomendación.
- Colaboración en la documentación del apartado 4.2 Implementación del *back-end*.
- Realización del diagrama de módulos del *front-end* (en el apartado 4.3)
- Documentación del desarrollo asociado a los dispositivos Beacon en el *front-end*.
- Documentación de las pruebas de arquitectura realizadas durante etapas anteriores (en el apartado 4.5).
- Colaboración en la planificación de la evaluación de usuarios (en el apartado 5.1).
- Realización y explicación del diagrama cronológico del proyecto (en el capítulo 8).
- Colaboración en la redacción del capítulo 8 Organización del trabajo.
- Colaboración en la documentación de los servicios de la API del ANEXO I.
- Búsqueda de información y referencias que apoyen los datos y tecnologías mencionados en este documento.
- Lectura de todo el documento y revisión del formato y contenidos del mismo.
- Numeración de los distintos capítulos y apartados del documento.

Además, mi aportación en cuestiones generales ha consistido en:

- Colaboración en la toma de decisiones.

- Asistencia a todas las reuniones quincenales con los tutores, tomando notas sobre los diferentes aspectos tratados en cada una de ellas y participando activamente.
- Comunicación con los tutores, mediante correo electrónico o tutorías presenciales, para la resolución de dudas o el intercambio de recursos necesarios para el desarrollo del proyecto.

9.2 Ignacio Rocillo Landa

Mi aportación en la etapa de investigación ha consistido en:

- Investigación sobre las distintas tecnologías de RA, con el fin de elegir aquella que más se adaptara a nuestras necesidades.
- Colaboración en la aportación de ideas y posterior definición de la funcionalidad de la aplicación.
- Desarrollo de aplicaciones de prueba utilizando la tecnología Unity + Vuforia. Ya había utilizado Unity y C# en la carrera con anterioridad y no me resultó difícil de manejar. Sin embargo, Vuforia era una tecnología totalmente nueva y tuve que aprender a utilizarla para realizar las pruebas. Las pruebas que realicé fueron:
 - Prueba de reconocimiento de texto con Vuforia.
 - Prueba de reconocimiento de imágenes en la nube con Vuforia.
- Colaboración en la definición de los servicios de la API.
- Realización de tutoriales para aprender a utilizar el servidor de aplicaciones, Tomcat, y la herramienta para la gestión y construcción de proyectos Java, Maven, aunque luego terminara realizando la parte de cliente (*front-end*).
- Investigación sobre las distintas fuentes de información de películas que nuestra aplicación podría hacer uso.
- Investigación de implementación del GPS en aplicaciones Android y de aplicaciones que utilizan esta tecnología.
- Investigación en la resolución de algunas cuestiones con la tecnología de RA Wikitude, tales como realizar llamadas entre Java y JavaScript en un mismo proyecto, botones y videos, etc.
- Estudio y comprensión de Wikitude mediante los *samples* que hay a disposición de los usuarios.

En la etapa de desarrollo:

- Diseño y desarrollo de la RA y de todas sus funcionalidades, las cuales son:
 - Mostrar video de la película.
 - Mostrar información de la película.
 - Valorar la película.
 - Añadir película a lista de deseos.
 - Compartir película en Twitter.
 - Mostrar amigos cercanos, valoración de amigos cercanos y puntuación media.
- Colaboración en la integración de la funcionalidad de los *beacons* en el proyecto.

- Colaboración en la integración de la aplicación Android.
- Desarrollo de la demo de la aplicación.

Mi aportación en la etapa de memoria ha consistido en:

- Documentación de las funcionalidades, las que corresponden a lo que he hecho en la etapa de desarrollo y de Funciones de la Aplicación.
- Colaboración en la documentación de los Estados del Arte, en los apartados de RA y Fuentes de Información de películas.
- Colaboración en la documentación de la implementación:
 - Colaboración en la documentación de Tecnologías usadas en la implementación.
 - Colaboración en la documentación de algunas Herramientas usadas en la implementación en común.
- Colaboración en la planificación de Evaluación de Usuarios y creación de los formularios a rellenar por el moderador y el usuario en la Evaluación de Usuarios.
- Generación de encuestas en Evaluación de Usuarios.
- Documentación y estudio de las conclusiones sacadas tras la Evaluación de Usuarios.
- Documentación de las pruebas realizadas durante etapas anteriores.
- Lectura de todo el documento y revisión de los contenidos del mismo.
- Corrección de la redacción de los apartados de Introducción, Conclusión, Trabajo Futuro, Pruebas e Implementación de GPS.

Además, mi aportación en cuestiones generales ha consistido en:

- Colaboración en la toma de decisiones.
- Asistencia a prácticamente todas las reuniones quincenales con los profesores, participando en ellas.

9.3 Jorge Muñoz Rodríguez

Mi aportación en la etapa de investigación ha consistido en:

- Investigación sobre las distintas tecnologías de RA, con el fin de elegir aquella que más se adaptara a nuestras necesidades.
- Colaboración en la aportación de ideas y posterior definición de la funcionalidad de la aplicación.
- Investigación sobre google material design para la creación de una interfaz.
- Investigación sobre el funcionamiento de los dispositivos Beacon y su implementación en Java con el SDK para Android de Konkakt.io

- Investigación sobre el ecosistema alrededor de los dispositivos Beacon.
- Realización de tutoriales de Android Studio.
- Realización de tutoriales de Unity.
- Colaboración en la definición de los servicios de la API.
- Realización de primeras interfaces de la aplicación.

En la etapa de desarrollo:

- Mi aportación ha sido en el front-end.
 - Realización de la interfaz de gente cercana.
 - Creación de los servicios de geolocalización.
 - Colaboración en el desarrollo de la implementación de los dispositivos Beacon.
 - Colaboración en la integración de la aplicación Android.
 - Desarrollo de la demo de la aplicación.

Mi aportación en la etapa de memoria ha consistido en:

- Colaboración en la documentación de los servicios de la API.
- Documentación de las funcionalidades, las que corresponden a lo que he hecho en la etapa de desarrollo.
- Colaboración en la documentación de la implementación.
 - Colaboración en la documentación de Tecnologías usadas en la implementación.
 - Explicación de los diagramas de E-R y las colecciones de mongo.
- Agregar una introducción en todos los apartados que era necesario.
- Agregar tabla de referencias y de figuras.
- Redacción en los apartados de Introducción, Conclusiones y Trabajo futuro.
- Estado del arte de los sistemas de localización.
- Documentación de las pruebas realizadas de Geolocalización.
- Colaboración en la Evaluación de usuario.

Además, mi aportación en cuestiones generales ha consistido en:

- Colaboración en la toma de decisiones.
- Asistencia a prácticamente todas las reuniones quincenales con los profesores, participando en ellas.

9.4 Jorge Rueda Garzón

Mi aportación en la etapa de investigación ha consistido en:

- Investigación sobre las distintas tecnologías de RA, con el fin de elegir aquella que más se adaptara a nuestras necesidades.
- Colaboración en la aportación de ideas y posterior definición de la funcionalidad de la aplicación.
- Pruebas de Wikitude con AndroidStudio y Eclipse
- Investigación sobre instalación y configuración de Tomcat en servidores Linux,
- Investigación sobre instalación, configuración y uso de Mysql server en una maquina con Ubuntu.
- Investigación sobre instalación, configuración y uso de MongoDB en una maquina con Ubuntu.
- Investigación sobre el uso de MongoDb y la geolocalización
- Investigación sobre Hibernate y tablas con claves compuestas.
- Investigación sobre el uso de un servidor de manera local con un dispositivo Android.
- Investigación sobre el uso de Maven y resolución de dependencias en las librerías.
- Investigación sobre la creación de servicios de arquitectura REST con Jersey.
- Investigación sobre el funcionamiento de los dispositivos Beacon y su implementación en Java con el SDK para Android de Konkakt.io
- Investigación del funcionamiento de la plataforma JColibrí
- Investigación y desarrollo de pruebas con la API de Facebook.
- Investigación sobre las distintas APIs disponibles para obtener información de películas

En la etapa de desarrollo:

- Mi aportación ha sido en el *back-end*:
 - Desarrollo de la API REST de servicios, junto con Cristina Delgado.
 - Realización del servicio de prueba de publicación en Facebook.
 - Configuración y mantenimiento del servidor proporcionado por la Universidad
 - Instalación y configuración de todo lo necesario en el Servidor
 - Creación y configuración de las bases de datos en el servidor.
 - Despliegue de la aplicación

Mi aportación en la etapa de memoria ha consistido en:

- Colaboración en la documentación de los servicios de la API.
- Realización del diagrama Entidad-Relación de la base de datos MySQL.
- Documentación de la estructura de la base de datos de MongoDB.

- Colaboración en la documentación del apartado de Implementación del *back-end*.
- Documentación de las pruebas con Facebook.

Además, mi aportación en cuestiones generales ha consistido en:

- Colaboración en la toma de decisiones.
- Asistencia a prácticamente todas las reuniones quincenales con los profesores, participando en ellas.

9.5 Sergio Fuentes Urabayan

Mi aportación en la etapa de investigación ha consistido en:

- Investigación sobre las distintas tecnologías de RA, con el fin de elegir aquella que más se adaptara a nuestras necesidades.
- Colaboración en la aportación de ideas y posterior definición de la funcionalidad de la aplicación.
- Desarrollo de aplicaciones básicas de prueba. Sobre todo aplicaciones Android que me sirvieron para aprender. Por ejemplo: login, registro, uso de *fragmentos*, creación de IU, uso servicios asíncronos, etc.
- Investigación sobre el uso de servicios REST desde Java Android. Uso de diferentes métodos de llamadas a los servicios.
- Colaboración en la definición de los servicios de la API.
- Realización de tutoriales para aprender a crear diseños de las interfaces con XML.
- Realización de tutoriales para saber cómo organizar de forma eficiente el código en Android Studio.
- Realización de tutoriales para configurar los *manifest* y usar Gradle en Android Studio.
- Creación de prototipos para el diseño de la aplicación.
- Primeros diseños de las interfaces de usuario utilizando diferentes herramientas, así como realizar tutoriales para aprender a usar dichas herramientas.
- Investigación sobre el funcionamiento de distintos tipos de desarrollo Android, utilizando librerías y proyectos en GitHub.

En la etapa de desarrollo:

- Diseño de todas las IU, exceptuando la de RA. Esto incluye:
 - Interfaz de Login
 - Interfaz de Registro
 - Interfaz de Inicio
 - Barra de Menú
 - Interfaz de Me Gusta
 - Interfaz de la Ficha de Película
 - Interfaz de Gente Cercana
- Desarrollo de las siguientes funcionalidades:
 - Login y Registro
 - Inicio y Menú
 - Mostrar información de la película seleccionada (Ficha de Película).
 - Mostrar las películas que le gustan al usuario (Me gusta).

- Desarrollo de las clases principales de la aplicación, como Usuario, Película, etc.
- Colaboración en la integración de la aplicación Android.
- Desarrollo de la demo de la aplicación.

Mi aportación en la etapa de memoria ha consistido en:

- Documentación de las funcionalidades, las que corresponden a lo que he hecho en la etapa de desarrollo.
- Colaboración en la documentación de la implementación de *front-end*
 - Colaboración en la documentación de Tecnologías usadas en la implementación.
 - Redacción del apartado Herramientas usadas en la implementación.
 - Redacción del apartado Entorno de desarrollo.
- Colaboración en la redacción del apartado de Organización del trabajo.
 - Documentación del apartado Herramientas de gestión de tareas.
 - Colaboración en la documentación del apartado Herramientas de control de versiones.
- Colaboración en la Evaluación de usuario.
- Generación de la encuesta para la evaluación de usuario.
- Redacción del apartado de Primeros prototipos
- Revisión y mejora del apartado de Implementación front-end.
- Redacción de el resumen y el abstract.
- Redacción de la conclusión en inglés.

Además, mi aportación en cuestiones generales ha consistido en:

- Colaboración en la toma de decisiones.
- Asistencia a las reuniones quincenales con los tutores, participando activamente.
- Investigación y proposición de nuevas herramientas para el trabajo grupal, por ejemplo, la herramienta Trello.

REFERENCIAS

1. **Höllerer, Tobias H.** [En línea] http://web.cs.wpi.edu/~gogo/courses/imgd5100_2012f/papers/Hollerer_AR_2004.pdf.
2. **La Realidad Aumentada facilita la visita al yacimiento de la Villa Romana de l'Albir.** [En línea] <http://www.canalpatrimonio.com/la-realialidad-aumentada-facilita-la-visita-al-yacimiento-de-la-villa-romana-de-lalbir/>.
3. **Bichlmeier, Christoph.** Medical Augmented Reality. [En línea] <http://medicaugmentedreality.com/2016/02/helping-autists-to-interact-with-their-social-environment/>.
4. **Dequidt, Jeremie.** Youtube. [En línea] <https://www.youtube.com/watch?v=i6UrvuPPATk>.
5. **Using an iPhone and augmented reality to teach medical students.** [En línea] <http://www.imedicalapps.com/2014/04/iphone-augmented-reality-medical-students/>.
6. **Google glass for war: The US military funded smart helmet that can beam information to soldiers on the battlefield.** [En línea] <http://www.dailymail.co.uk/sciencetech/article-2640869/Google-glass-war-US-military-reveals-augmented-reality-soldiers.html>.
7. **¡DESCUBRE EN TU MÓVIL LAS CULTURAS DE AMÉRICA!** [En línea] <http://www.mecd.gob.es/museodeamerica/espacio-interactivo/Tanto-que-disfrutar-jugando---/RACMA.html>.
8. **Sony.** Playstation. [En línea] <https://www.playstation.com/es-es/games/invizimals-ppsp/>.
9. **OpenCV.** [En línea] <http://opencv.org/>.
10. **Vuforia.** [En línea] <https://developer.vuforia.com/>.
11. **Metaio.** [En línea] <https://www.metaio.com/>.
12. **Mixare.** [En línea] <http://www.mixare.org/>.
13. **ARToolKit.** [En línea] <https://www.hitl.washington.edu/artoolkit/>.
14. **Wikitude.** [En línea] <http://www.wikitude.com/>.
15. **Metacritic.** [En línea] <http://www.metacritic.com/>.
16. **IMDb.** [En línea] <http://www.imdb.com/>.
17. **OMDb.** [En línea] <http://www.omdbapi.com/>.
18. **Tomatoes, Rotten.** [En línea] <http://www.rottentomatoes.com/>.

19. Rotten Tomatoes API. [En línea] <http://developer.rottentomatoes.com/>.
20. TheMovieDb.org. [En línea] <https://www.themoviedb.org/?language=es>.
21. TheMovieDb.org API. [En línea] <https://www.themoviedb.org/documentation/api>.
22. ¿Qué son los sistemas de recomendación? [En línea] <http://jarroba.com/que-son-los-sistemas-de-recomendacion/>.
23. LibRec. [En línea] <http://www.librec.net/>.
24. LensKit. [En línea] <http://lenskit.org/>.
25. Mahout. [En línea] <http://mahout.apache.org/>.
26. ounae. <http://ounae.com/aplicaciones-usos-beacons-bluetooth-emisores/>. [En línea]
27. Estimote Beacon. <http://estimote.com/>. [En línea]
28. RadBeacon. <http://www.radiusnetworks.com/>. [En línea]
29. Bluecat Beacon. <http://bluecats.com/>. [En línea]
30. Kontakt.io. <https://kontakt.io/>. [En línea]
31. Richardson Maturity Model. [En línea] <http://martinfowler.com/articles/richardsonMaturityModel.html>.
32. Hat Hexacta - Introducción a REST. [En línea] <http://hat.hexacta.com/introduccion-a-rest-22/>.
33. Asier Marques - Conceptos sobre APIs REST. [En línea] <http://asiermarques.com/2013/conceptos-sobre-apis-rest/>.
34. Creating a User-Based Recommender in 5 minutes. [En línea] <https://mahout.apache.org/users/recommender/userbased-5-minutes.html>.
35. MovieTweatings. [En línea] <https://github.com/sidooms/MovieTweatings>.
36. Konkakt.io - Android SDK Quickstart. [En línea] <http://developer.kontakt.io/android-sdk/2.1.0/quickstart/>.
37. Realidad aumentada con Unity 5. [En línea] <http://emiliusvgs.com/realidad-aumentada-con-unity-5/>.
38. How to Create a Simple Cloud Recognition App in Unity. [En línea] <https://developer.vuforia.com/library/articles/Solution/How-To-Create-a-Simple-Cloud-Recognition-App-in-Unity>.

39. Portal para desarrolladores de Vuforia. [En línea] <https://developer.vuforia.com/>.
40. Creando aplicaciones de Realidad Aumentada con Reconocimiento de Imágenes. [En línea] <http://www.desarrollolibre.net/blog/agrupados/creando-aplicaciones-de-realidad-aumentada-con-reconocimiento-de-imagenes/46>.
41. Configurar el entorno de trabajo JEE con Eclipse Kepler y Tomcat 7. [En línea] https://www.youtube.com/watch?v=4NcOjx40_do.
42. Tutorial de introducción a Maven 3. [En línea] http://static1.1.sqspcdn.com/static/f/923743/15025126/1320942755733/Tutorial_de_Maven_3_Erick_Camacho.pdf?token=bxGhbCQGvIEITO%2Fw6pK9Bz00iGY%3D.
43. Jersey hello world example. [En línea] <http://www.mkyong.com/webservices/jax-rs/jersey-hello-world-example/>.
44. Hibernate - Parte 2: Persistiendo Objetos Simples usando Anotaciones (metadatos). [En línea] <http://www.javatutoriales.com/2009/05/hibernate-parte-2-persistiendo-objetos.html>.
45. Get started quickly with Hibernate Annotations and JPA2. [En línea] <http://www.ocpssoft.org/java/getting-started-quickly-with-hibernate-annotations/>.
46. Twitter4J Code Examples. [En línea] <http://twitter4j.org/en/code-examples.html>.
47. Un poco de Twitter4J (Twitter + Java). [En línea] <https://unpocodejava.wordpress.com/2014/10/06/un-poco-de-twitter4j-twitter-java/>.
48. Tweet Button. [En línea] <https://dev.twitter.com/web/tweet-button>.
49. Gracia, Luis Miguel. unpocodejava.wordpress.com. *Un poco de Facebook4J (Facebook + Java)*. [En línea] 7 de Octubre de 2014. <https://unpocodejava.wordpress.com/2014/10/07/un-poco-de-facebook4j-facebook-java/>.
50. RACMA: Aplicación de Realidad Aumentada para el Museo de América. [En línea] <http://eprints.ucm.es/32915/1/Realidad%20aumentada%20para%20el%20Museo%20de%20Am%C3%A9rica.pdf>.
51. Xloudia. [En línea] <http://www.xloudia.com/>.
52. TheMovieDb.org. API TheMovieDb.org. [En línea] <https://www.themoviedb.org/documentation/api>.

ANEXOS

ANEXO I: API REST

Esquema:

Todo acceso a la API es a través de HTTP, y se hace desde el dominio `container.fdi.ucm.es:20041/MuviAppREST`. Todos los datos se envían y se reciben como JSON.

Las URLs de nuestra API, por tanto, tienen la siguiente estructura:

```
http://container.fdi.ucm.es:20041/ruta_del_recurso>?<consulta_de_filtrado>
```

1. Usuarios

Login usuario

Comprueba los datos de acceso de un usuario.

POST /usuarios/{alias_usuario}

Parámetros:

Nombre	Tipo	Descripción
password	string	La contraseña del usuario que desea acceder a la aplicación.

Ejemplo:

```
{
  "password": "p4ssw0rd"
}
```

Respuesta:

Status: 200 OK
<pre>{ "id": 1, "alias": "carmengon", "password": "p4ssw0rd", "nombre": "carmen gonzalez", "email": "carmengon@muvi.es" }</pre>

Si el usuario introducido no corresponde con ninguna cuenta:

```
Status: 422 Unprocessable Entity
```

Registro usuario

Registra una nueva cuenta de usuario.

POST /usuarios

Parámetros:

Nombre	Tipo	Descripción
alias	string	El nombre del usuario de la nueva cuenta.
password	string	La contraseña de la nueva cuenta.
nombre	string	El nombre de pila asociado a la nueva cuenta.
email	string	El correo electrónico asociado a la nueva cuenta.

Ejemplo:

```
{
  "alias": "carmengon",
  "password": "p4ssw0rd",
  "nombre": "carmen gonzalez",
  "email": "carmengon@muvi.es"
}
```

Respuesta:

Status: 201 Created

```
{
  "id": 1,
  "alias": "carmengon",
  "password": "p4ssw0rd",
  "nombre": "carmen gonzalez",
  "email": "carmengon@muvi.es"
}
```

Si ya hay un usuario registrado con el mismo alias:

Respuesta:

Status: 409 Conflict

```
{
  "mensaje": "nombre de usuario no disponible"
}
```

Si ya hay un usuario registrado con el mismo correo electrónico:

Respuesta:

Status: 409 Conflict

```
{
  "mensaje": "correo electrónico no disponible"
}
```

Buscar usuario

Busca un usuario por su nombre.

GET /usuarios?usuario=name

Respuesta:

Status: 200 OK
<pre>{ "id": 1, "alias": "carmengon", "password": "p4ssw0rd", "nombre": "carmen gonzalez", "email": "carmengon@muvi.es" }</pre>

Si el usuario introducido no corresponde con ninguna cuenta:

Status: 401 Unauthorized

2. Películas

Buscar información de película

Busca la información de una película por su título.

GET /peliculas?titulo={titulo}

Respuesta:

Status: 200 OK
<pre>{ "id_IMDB": 3077214, "id_TheMovieDB": 245168, "titulo": "Suffragette", "sinopsis": "Based on true events about the foot soldiers of the early feminist movement who were forced underground to evade the State.", "generos": ["Drama", "History"], "votacion": 6.9, "trailer": "https://www.youtube.com/watch?v=gYXfARbezcA" }</pre>

Nota: Si no se encuentra el tráiler de esa película, el campo `tráiler` será `null`. Si no tiene `id_IMDB`, será 0.

Si no se encuentra ninguna película que coincida con ese título:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "película no encontrada" }

Listar películas

Lista las películas que coincidan con un título. El resultado está limitado a 5 películas.

GET /peliculas/list?titulo={titulo}

Respuesta:

Status: 200 OK
{ "id_IMDB": 3077214, "id_TheMovieDB": 245168, "titulo": "Suffragette", "sinopsis": "Based on true events about the foot soldiers of the early feminist movement who were forced underground to evade the State.", "generos": ["Drama", "History"], "votacion": 6.9, "trailer": "https://www.youtube.com/watch?v=gYXfARbezCA" }

Nota: Si no se encuentra el tráiler de esa película, el campo `tráiler` será `null`. Si no tiene `id_IMDB`, será 0.

Si no se encuentra ninguna película que coincida con ese título:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "película no encontrada" }

3. Valoraciones

Añadir valoración

Añade una valoración de una película realizada por un usuario.

POST /valoraciones

Parámetros:

Nombre	Tipo	Descripción
id_usuario	long	El identificador del usuario que realiza la valoración.
id_pelicula	long	El identificador de la película valorada.
valoración	int	La valoración realizada de la película

Ejemplo:

```
{
  "id_usuario": 1,
  "id_pelicula": 1,
  "valoracion": 7
}
```

Respuesta:

Status: 201 Created
{ "id_usuario": 1, "id_pelicula": 1, "valoracion": 7 }

Si ese usuario ya ha valorado esa película:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "ese usuario ya ha valorado esa película" }

Mostrar valoración

Muestra la valoración de una película realizada por un usuario.

GET /valoraciones?usuario={id_usuario}&pelicula={id_pelicula}

Respuesta:

Status: 200 OK
<pre>{ "id_usuario": 1, "id_pelicula": 1, "valoracion": 7 }</pre>

Si ese usuario no ha valorado esa película:

Respuesta:

Status: 422 Unprocessable Entity
<pre>{ "mensaje": "ese usuario no ha valorado esa película" }</pre>

4. Deseos

Añadir deseo

Añade una película a la lista de deseos de un usuario.

POST /deseos

Parámetros:

Nombre	Tipo	Descripción
id_usuario	long	El identificador del usuario que añade el deseo.
id_pelicula	long	El identificador de la película que se añade como deseo.

Ejemplo:

<pre>{ "id_usuario": 1, "id_pelicula": 20 }</pre>

Respuesta:

Status: 201 Created
<pre>{ "id_usuario": 1, "id_pelicula": 20 }</pre>

Si ese deseo ya existe:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "ese deseo ya existe" }

Listar deseos

Lista las películas de la lista de deseos de un usuario.

GET /deseos/{id_usuario}

Respuesta:

Status: 200 OK
[{ "id": 20, "título": "Star Wars VII: El despertar de la fuerza" }]

Buscar deseo

Muestra la información de una película de la lista de deseos del usuario.

GET /deseos?usuario={id_usuario}&pelicula={id_pelicula}

Respuesta:

Status: 200 OK
{ "resultado": "true" }

Eliminar deseo

Elimina una película de la lista de deseos de un usuario.

Nota: No se elimina la película, sólo la elimina de la lista de deseos.

DELETE /deseos?usuario={id_usuario}&pelicula={id_pelicula}

Respuesta:

Status: 204 No content

Si la película no existe en la lista de deseos del usuario:

Respuesta:

Status: 409 Conflict
{ "mensaje": "ese deseo no existe" }

5. Recomendaciones

Estimar recomendación

Estima lo que una película le gustaría a un usuario.

POST /recomendador?usuario={id_usuario}&pelicula={id_pelicula}

Respuesta:

Status: 200 OK
{ "estimacion": "7" }

6. Geolocalización (GPS)

Añadir ubicación de usuario

Añade la ubicación de un usuario.

POST /geolocalización

Parámetros:

Nombre	Tipo	Descripción
id_usuario	long	El identificador del usuario al que pertenece la ubicación.
latitud	double	La latitud de la ubicación del usuario.
longitud	double	La longitud de la ubicación del usuario.

Ejemplo:

{ "id_usuario": 1, "latitud": 40.463576, "longitud": -3.616432 }
--

Respuesta:

Status: 200 OK
<pre>{ "id_usuario": 1, "name": "Jorge", "loc": { "longitud": -3.616432, "latitud": 40.463576 } }</pre>

Eliminar ubicación de usuario

Elimina la ubicación de un usuario.

```
DELETE /geolocalizacion?id_usuario={id_usuario}
```

Respuesta:

Status: 204 No Content

Si no existe la ubicación del usuario en el sistema:

Respuesta:

Status: 422 Unprocessable Entity
<pre>{ "mensaje": "no se conoce la ubicación de ese usuario" }</pre>

Listar usuarios cercanos

Lista los usuarios cercanos con respecto a la ubicación actual.

```
GET /geolocalizacion?lat={latitud}&lon={latitud}&max{distanciaMax}
```

Respuesta:

Status: 200 OK
<pre>{ "id_usuario": 1 "name": "Jorge", "distancia": 200 }</pre>

7. Beacons

Asociar usuario a beacon

Asocia un usuario a un beacon cuando se conecta a dicho beacon.

POST /beacons

Parámetros:

Nombre	Tipo	Descripción
id_usuario	long	El identificador del usuario que realiza el evento.
id_beacon	long	El identificador del beacon.

Ejemplo:

```
{
  "id_usuario": 1,
  "id_beacon": "AB123123"
}
```

Respuesta:

Status: 200 OK
{ "id_usuario": 1, "id_beacon": "AB123123", "name": "carmen gonzalez" }

Si el usuario no existe en el sistema:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "ese usuario no existe" }

Eliminar usuario de beacon

Elimina un usuario de un beacon una vez ha pasado cierto tiempo desconectado.

DELETE /beacons?usuario={id_usuario}&beacon={id_beacon}

Respuesta:

Status: 204 No Content

Si el usuario no existe en el sistema:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "ese usuario no existe" }

Si el usuario no existe en el beacon:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "ese usuario no existe en ese beacon" }

Listar usuarios de beacon

Lista los usuarios conectados a un mismo beacon.

GET /beacons?beacon={id_beacon}

Respuesta:

Status: 200 OK
[{ "id_usuario": 1, "id_beacon": 504, "nombre": "asdasd" }, { "id_usuario": 1, "id_beacon": 504, "nombre": "afdsfsd" }]

Si el beacon no tiene usuarios:

Respuesta:

Status: 422 Unprocessable Entity
{ "mensaje": "ese beacon no tiene usuarios" }